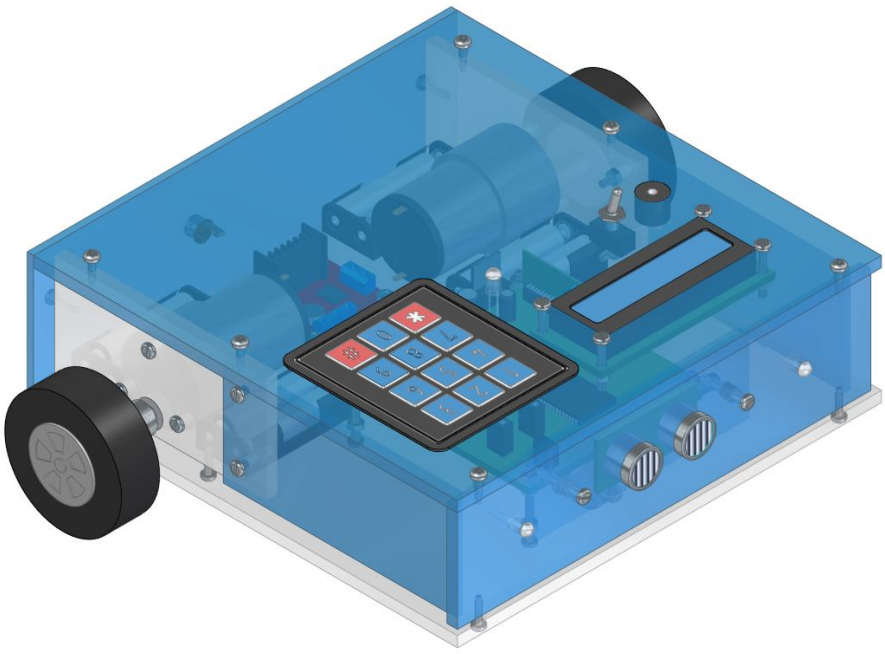
DISEÑO Y CONSTRUCCIÓN DE UN PROTOTIPO ROBÓTICO PARA TRANSPORTE

CARLOS ALBERTO GOYENECHE ALFONSO



UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA
ESCUELA DE CIENCIAS BÁSICAS TECNOLOGÍAS E INGENIERÍA
CEAD SOGAMOSO

2016

DISEÑO Y CONSTRUCCIÓN DE UN PROTOTIPO ROBÓTICO PARA TRANSPORTE

Alumno:
CARLOS ALBERTO GOYENECHE ALFONSO
Código: 79941769
agc.control@gmail.com

PROYECTO PARA OBTENER EL TITULO DE INGENIERO ELECTRÓNICO.

Presentado al profesor:
OSCAR IVÁN VALDERRAMA ARIAS
Director

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA
ESCUELA DE CIENCIAS BÁSICAS TECNOLOGÍAS E INGENIERÍA
CEAD SOGAMOSO

2016

DEDICATORIA

Muy especial, a mi esposa María Claudia Mendoza Quezada, quien con su amor y apoyo incondicional me acompaña en la ardua tarea de vivir, ella me motiva a lograr mi sueño, a ser un mejor ser humano, me quiere y me respeta.

A mis hermosos e inteligentes hijos, Cristian Zamir Goyeneche Mendoza y María Zaray Goyeneche Mendoza, los dos motores de mi existencia, ellos son mi más grande tesoro, la mayor fuente de alegrías y satisfacciones. Son los diseñadores de mi vida, los ingenieros de felicidad. Es por ello que he realizado mi estudio de ingeniería electrónica y son ellos con su cariño los que me han fortalecido en los días difíciles.

A la memoria de mi hermano Oscar Javier.

AGRADECIMIENTO

Muy especial, a la universidad UNAD, por la labor social de impartir conocimiento y de fortalecer a las personas que toman la decisión de estudiar y de salir adelante. Por aceptar mi solicitud de prepararme como profesional y permitirme desarrollar mi sueño de temprana edad.

Al ingeniero Oscar Iván Valderrama, por su ayuda en el transcurso de mi carrera, por creer en mi trabajo y su apoyo incondicional. Por otorgarme su confianza y resaltar mi dedicación.

A la excelentísima ingeniera Sandra Isabel Vargas, por sus explicaciones y la confianza que siempre me otorgó.

Al ingeniero Juan Olegario Monroy, por su ayuda y sus consejos, en la construcción de este documento y la mejora de mi proyecto.

Al más grande científico e ingeniero, Nikola Tesla, que aunque solo esté presente en algunos libros, es para mí ejemplo de trabajo, dedicación y amor por la naturaleza. Sin su aporte a la ciencia, la vida no sería lo que conocemos, pues aprendí en mis estudios, que él fue la persona con más aportes al desarrollo de la electricidad, y sus desarrollos son la base de la electrónica.

CONTENIDO

INTRODUCCIÓN ---- 15

1. MARCO DE REFERENCIA ---- 18

1.1. PLANTEAMIENTO DEL PROBLEMA ---- 18

1.2. OBJETIVOS ---- 22

1.3. JUSTIFICACIÓN ---- 23

1.4. ALCANCES DEL PROYECTO ---- 24

1.5. MEJORAS DEL PROYECTO ---- 25

2. MARCO TEÓRICO ---- 26

2.1. ESTRUCTURAS BÁSICAS. ---- 27

2.1.1. Grados de libertad ---- 29

2.1.2. Modelado de la cinemática ---- 30

2.2. SISTEMA ELÉCTRICO DE ALIMENTACIÓN ---- 32

2.2.1. Pilas de Níquel Cadmio ---- 32

2.2.2. Acondicionamiento del voltaje ---- 34

2.3. ELEMENTOS DE ENTRADA ---- 35

2.3.1. Sensor infrarrojo ---- 36

2.3.2. Sensor por Ultrasonido ---- 38

2.3.3. Teclado matricial ---- 42

2.3.4. Modulo RF M40 RFC ---- 43

2.4. ELEMENTO CONTROLADOR ---- 44

2.4.1. Microcontrolador PIC16F877A ---- 44

2.4.2. Conexión básica del microcontrolador ---- 47

2.4.3. Instrucciones del microcontrolador ---- 49

2.4.4. Consideraciones para la programación ---- 51
2.5. ELEMENTOS DE SALIDA ---- 53
2.5.1. Motor de corriente continúa ---- 53
2.5.2. Diodo LED ---- 56
2.5.3. Pantalla LCD 2x16 ---- 58
3. DISEÑO Y DESARROLLO DEL PROYECTO ---- 60
3.1. ESTRUCTURA MECÁNICA ---- 61
3.1.1. Elaboración de la plataforma ---- 62
3.2. CIRCUITO ELECTRÓNICO ---- 71
3.2.1. Esquema electrónico en Proteus ISIS ---- 74
3.2.2. Descripción del circuito electrónico ---- 81
3.2.3. Diseño del PCB en ARES de Proteus ---- 84
3.2.4. Construcción de la tarjeta PCB ---- 92
3.2.5. Técnica para soldar y fijar los componentes ---- 95
3.3. PROGRAMACIÓN DEL PROTOTIPO ROBOT ---- 98
3.3.1. Pruebas con MPLAB ---- 98
3.3.2. Programación en CCS ---- 109
3.3.3. Grabación en el microcontrolador ---- 123
3.3.4. Programa final ---- 126
4. USO Y MANTENIMIENTO ---- 134
4.1. GUIA DE USO DEL PROTOTIPO ---- 134
4.2. MANTENIMIENTO DEL PROTOTIPO ---- 138
4.2.1. Limpieza ---- 138
4.2.2. Inspección ---- 139
4.2.3. Ajuste ---- 140

5. ANÁLISIS ECONÓMICO ---------- 143

5.1. EQUIPOS Y COMPONENTES ---------- 143

5.2. PRESUPUESTO ---------- 143

6. CONCLUSIONES ---------- 146

BIBLIOGRAFÍA ---------- 148

ANEXOS ---------- 149

LISTA DE FIGURAS

Figura 1. Esquema trazado de la línea de ensamble. 21
Figura 2. Estructura diferencial. 27
Figura 3. Estructura síncrono 27
Figura 4. Estructura triciclo. 28
Figura 5. Estructura carro. 28
Figura 6. Grafica de cinemática de la plataforma triciclo. 29
Figura 7. Fotografía de las baterías AAA 32
Figura 8. Fotografía sensores reflectivos. 36
Figura 9. Esquema del sensor reflectivo. 37
Figura 10. Esquema funcionamiento básico. 38
Figura 11. Fotografía sensor de ultrasonido. 41
Figura 12. Fotografía del teclado matricial. 42
Figura 13. Diagrama de pines microcontrolador. 45
Figura 14. Fotografía circuito integrado PIC16F877A. 47
Figura 15. Esquema circuito básico del PIC. 48
Figura 16. Fotografía del motor DC con reductor de velocidad. 53
Figura 17. Fotografía de los diodos LED. 56
Figura 18. Esquema circuito diodo LED. 57
Figura 19. Fotografía de la pantalla LCD 2X16. 58
Figura 20. Dibujo de la plataforma vista por debajo. 61
Figura 21. Dibujo de la plataforma vista de lado derecho. 62
Figura 22. Fotografía del acrílico pieza A1. 65
Figura 23. Fotografía de los soportes pieza A2. 65

Figura 24. Fotografía de soportes terminados. 66
Figura 25. Fotografía motoreductor en la base. 66
Figura 26. Fotografía de la base A terminada. 67
Figura 27. Fotografía de la pieza frontal B2. 67
Figura 28. Fotografía de la pieza B1. 68
Figura 29. Fotografía de instalación pieza B1. 69
Figura 30. Fotografía de la plataforma. 69
Figura 31. Dibujos de las piezas que forman a estructura. 70
Figura 32. Pines de salida programador Pickit 2. 71
Figura 33 Esquema microcontrolador conexionado a grabador. 72
Figura 34. Esquema del circuito LCD y teclado. 74
Figura 35. Icono de Proteus en escritorio. 75
Figura 36. Ventana entorno de trabajo en Proteus. 75
Figura 37. Submenús de trabajo Proteus. 76
Figura 38. Selección componentes en Proteus 76
Figura 39. Ventana edición componentes. 77
Figura 40. Lista de componentes añadidos. 77
Figura 41. Menú del componente activado 78
Figura 42. Esquema del circuito con componentes unidos 79
Figura 43. Ventana de edición de componente 79
Figura 44. Esquema del circuito electrónico para simulación. 80
Figura 45. Diagrama de bloques sistema electrónico 81
Figura 46. Fotografía del circuito cableado en protoboard. 83
Figura 47. Ventana de edición componentes en ARES. 84
Figura 48. Esquema con cambios para exportar a ARES 85

Figura 49. Icono para transferencia a ARES. 85
Figura 50. Ventana de trabajo en ARES 86
Figura 51. Ventana de trabajo en ARES 86
Figura 52. Área de la PCB en ARES 87
Figura 53. Menu Tolls opción Auto Placer. 87
Figura 54 Ventana de opciones de pegado en ARES. 88
Figura 55. Ubicación con Auto Placer en ARES. 88
Figura 56. Ventana de funciones para los componentes en ARES. 89
Figura 57. Posición de los componentes en ARES. 89
Figura 58. Menú Tools opción Auto Router. 90
Figura 59. Ventana de propiedades de Auto Router. 90
Figura 60. Vista del diseño de la PCB en ARES 91
Figura 61. Menú Output opción 3D Visualization. 91
Figura 62. Tarjeta PCB en 3D con ARES. 92
Figura 63. Fotografía tarjeta PCB terminada 94
Figura 64. Fotografía de la tarjeta PCB con componentes montados. 96
Figura 65. Fotografía de la tarjeta PCB montada en la plataforma. 97
Figura 66. Diagrama de decisiones para el primer algoritmo. 98
Figura 67. Icono Mplab en escritorio. 99
Figura 68. Ventana de trabajo de Mplab 100
Figura 69. Barra de botones para simulación de Mplab. 103
Figura 70. Escritorio de Mplab para simulación. 103
Figura 71. Ventana de entorno en Proteus ISIS para simulación. 104
Figura 72. Ventana de propiedades del microcontrolador en ISIS. 104
Figura 73. Diagrama de decisiones para segundo algoritmo. 105

Figura 74. Icono PIC C en escritorio. 109
Figura 75. Ventana de entorno CCS PCW. 110
Figura 76. Icono de PIC Wizard.....110
Figura 77. Ventana para guardar el PIC Wizard. 111
Figura 78. Ventana general PIC Wizard. 111
Figura 79. Ventana de comunicaciones.....112
Figura 80. Ventana de entradas análogas.....112
Figura 81. Ventana ara otros recursos.....113
Figura 82. Ventana Interrupciones. 113
Figura 83. Ventana Drivers. 114
Figura 84. Ventana configuración de pines. 114
Figura 85. Presentación en la pantalla LCD de la rutina Inicial. 116
Figura 86. Presentación en la pantalla LCD de la rutina clave. 117
Figura 87. Icono de PICkit 2 en el escritorio.....123
Figura 88. Ventana de edición Pickit2.....124
Figura 89. Panel frontal del prototipo. 134
Figura 90. Fotografía del control remoto.....136
Figura 91. Dibujos procedimiento de inspección. 142

LISTA DE TABLAS

Tabla 1. Instrucciones orientadas a los bytes. 49

Tabla 2. Instrucciones orientadas a los bits. 50

Tabla 3. Instrucciones con literales. 50

Tabla 4. Costo de componentes electrónicos. 144

Tabla 5. Costo de materiales. 145

Tabla 6. Costos totales del proyecto. 145

ABSTRACT

The transport of objects is a basic necessity of daily life; with the passage of time techniques and methods have been developed to make this activity easier for humans. One of the innovations that stand out today is the devices called Robots.

The project seeks to put in context: kinematics in mini robots, mechanical parts elaboration, concepts of micro controlled electronics and programming, CAD management for circuit simulation and PCB card development and finally the construction of the control program.

This document contains the design and construction of a prototype robot, which describes the electronic components used; these are classified into input elements, control element and output elements. It shows the elaboration of the mechanical parts, with drawings and illustrations and the spatial location. The sensors are located inside the platform in order to build the guidance system.

The construction of the electronic circuit is shown starting with the outline, the design of the printed card, and the construction of the same; in its final part is written the control program called "test program" and the program in language C as "final program" of this project.

The robot prototype, as an experimentation tool, allows the collection of information that leads to the design of a prototype of greater functionality, which is better suited to an industrial working environment. The results serve to refer to the quality of the mechanical and electronic components, the electronic functional characteristics and the agility of the recorded algorithm.

RESUMEN

El transporte de objetos es una necesidad básica de la vida cotidiana; con el pasar del tiempo se han desarrollado técnicas y métodos para hacer esta actividad más fácil para los seres humanos. Una de las innovaciones que se destacan en la actualidad son los aparatos llamados Robots.

El proyecto busca poner en contexto: la cinemática en los mini robots, elaboración de piezas mecánicas, conceptos de electrónica micro controlada y programación, manejo de CAD para simulación de circuitos y elaboración de tarjetas PCBs y por último la construcción del programa de control.

El presente documento contiene el diseño y la construcción de un prototipo robot, el cual describe los componentes electrónicos usados, estos se clasifican en elementos de entrada, elemento de control y elementos de salida. Se muestra la elaboración de las piezas mecánicas, con dibujos e ilustraciones y la ubicación espacial. Se ubican los sensores dentro de la plataforma con la finalidad de construir el sistema de guía.

Se muestra la construcción del circuito electrónico iniciando con el esquema, el diseño de la tarjeta impresa, la construcción de la misma; en su parte final se realiza la escritura del programa de control denominado: “programa de prueba” y el programa en lenguaje C, como “programa final” de este proyecto.

El prototipo robot, como herramienta de experimentación, permite recoger información que lleve a diseñar un prototipo de mayor funcionalidad, que se acomode mejor a un entorno de trabajo industrial. Los resultados sirven para referenciar la calidad de los componentes mecánicos y electrónicos, las características funcionales electrónicas y la agilidad del algoritmo grabado.

INTRODUCCIÓN

El presente proyecto tiene la finalidad de mostrar el diseño y construcción paso a paso de un prototipo robots, el cual va a ser utilizado en la actividad de transporte a nivel experimental, para posteriormente poder evaluar la viabilidad de construir un robot funcional que se ajuste a las necesidades industriales.

Es a partir de la observación en ambientes laborales de índole industrial, que se hace necesario la movilización de elementos tanto para la construcción y mantenimiento de maquinaria, se plantea un supuesto de línea de ensamblaje donde se evidencia la necesidad de un medio o elemento que desempeñe la función de transporte.

Surge la idea de diseñar y construir un prototipo para usar en tareas repetitivas relativamente simples, pero tediosas para los seres humanos. Se busca que este diseño imite las actuaciones humanas de manera automática.

El prototipo robot de este proyecto, es destinado a ser una herramienta de transporte experimental, inicialmente se ha tomado la idea de transportar tornillos pensando en futuros escenarios de manufactura, ya que los tornillos son piezas fundamentales en la unión de partes mecánicas, forman parte esencial en la construcción de casi todos los aparatos modernos.

En el trabajo escrito no se profundizará en la implementación del prototipo robot, aunque se tienen en cuenta las pruebas de movimiento cinemático, seguimiento de línea y detección de objetos, realizadas luego de su construcción, y en base a estas pruebas se desarrolla el programa de control final, uniendo todos los algoritmos. Del supuesto escenario de trabajo industrial se obtiene el diseñó del tablero de comando del prototipo o panel frontal, y basado en este se describe el modo de uso, cuidados y mantenimiento.

En el diseño del prototipo robot se tienen en cuenta, el entorno donde debe desplazarse, es decir el área de trabajo y se determina el sistema mecánico a usar o plataforma. Luego se construye dicha plataforma, y basándose en ella se realiza la ubicación espacial de los sensores, y se decide el circuito electrónico de control.

Una vez construida la plataforma se puede verificar el circuito electrónico y ver el comportamiento cinemático del prototipo, es aquí donde se realizan los cambios pertinentes para mejorar el diseño electrónico; se opta por el uso de un microcontrolador como cerebro del prototipo robot, por que en este es posible programar diferentes formas de desplazamiento, solo con cambiar el algoritmo de control.

Además de los elementos descritos, se tienen en cuenta dentro del programa de control cinemático, las características mecánicas del prototipo, la eficiencia de los actuadores, la velocidad de respuesta de los sensores, la velocidad de procesamiento del controlador y por último el programa desarrollado.

Aunque en la elaboración de un proyecto se consideren la mayoría de los escenarios técnicos, es posible que las cosas no resulten como se espera, y se debe estar sujeto a cambiar los diseños y en ocasiones reiniciar los trabajos de construcción; así que una de las ventajas de usar un circuito programable es facilitar dichos cambios.

El documento se ha organizado para presentar el diseño del prototipo robot de la siguiente forma:

- En el capitulo 1, se plantea el problema a resolver, el objetivo del proyecto, la justificación del proyecto, los alcances y mejoras futuras.

- En el capitulo 2, se abordan los conceptos necesarios que se deben conocer para construir el prototipo robot. Las estructuras mecánicas básicas, concepto de grados de libertad, el modelamiento cinemático. También en este capitulo se describe la fuente de alimentación eléctrica, se detalla el funcionamiento de los componentes electrónicos sensores, el teclado matricial, el modulo de radio frecuencia, el controlador con sus características básicas y consideraciones necesarias, la construcción interna del motor DC, se habla de los LEDs y las pantalla LCD.

- El capitulo 3, muestra la construcción en detalle del prototipo robot. Inicia con fabricación de la estructura mecánica o plataforma, luego se describe el

diseño del circuito electrónico y la tarjeta PCB usando Proteus, se muestra como se fabrica la tarjeta impresa pasando por detalles de elaboración y soldadura de los componentes. También se puede encontrar la construcción de los algoritmos de control y sus correspondientes programas en MPLAB y CCS de forma sencilla y bien detallada, se describe como grabarlos en el microcontrolador.

- El capitulo 4, aborda el uso del prototipo robot y mantenimiento, detalles como: limpieza, inspección y ajuste.

- El capitulo 5, presenta el análisis económico, los costos del proyecto y el presupuesto necesario para su construcción.

- En el capitulo 6, tienen las conclusiones obtenidas de todo el proceso de diseño y construcción del prototipo robot.

Finalmente, es importante resaltar que el prototipo robot de este proyecto hace parte de la robótica móvil, es decir maquinas capaces de trasladarse en su entorno. Se han convertido en un foco importante de la investigación actual y casi cada universidad importante que tenga un laboratorio centra la investigación en los robots móviles, buscando que sus estudiantes se interesen por el diseño y construcción de estas maquinas y participen en concursos de entretenimiento, por ejemplo: velocistas o seguidores de línea.

También encontramos empresas dedicadas a la construcción de este tipo de robots, como por ejemplo Robotnik, especializada en aplicaciones de plataformas móviles, en áreas de robótica profesional, robots civiles y agricultura. Cuentan con robots para la limpieza industrial, inspección y vigilancia, medición remota, seguridad, robots de defensa y rescate, manipuladores móviles para condiciones extremas, plataformas robóticas móviles para agricultura como su aplicación más innovadora. Recuperado de: http://www.robotnik.es

1. MARCO DE REFERENCIA

1.1. PLANTEAMIENTO DEL PROBLEMA

Un robot está compuesto por los siguientes elementos: estructura mecánica, trasmisiones, sistema de accionamiento, sistema sensorial, sistema de control y elementos terminales; estos elementos deben funcionar coordinadamente para que cumplan de manera eficaz la tarea proyectada del robot, es aquí que nace el concepto de cinemática, de vital importancia para llevar a cabo los movimientos que se desean obtener.

Para lograr poner en contexto la cinemática en los mini robots, elaboración de piezas mecánicas, conceptos de electrónica micro controlada y programación, manejo de CAD para simulación de circuitos, elaboración de tarjetas PCBs y la construcción del programa de control, fue necesario definir un escenario de aplicación del prototipo a diseñar; se plantea transportar tornillos, con un prototipo robot que use ruedas para su desplazamiento en una superficie plana, siendo guiado por línea trazada en dicha superficie. Teniendo en cuenta las extensas oportunidades en aplicaciones futuras, donde se ha tomado como elementos a transportar tornillos, por ser usados en innumerables aplicaciones de manufactura.

Para la cinemática, es necesario conocer y diferenciar entre las plataformas, ya que permite seleccionar la que mas se ajuste a la necesidad de trabajo. En el caso de este proyecto no se pretende una plataforma muy compleja, para lograr un ejemplo de desarrollo que integre todos los elementos antes expuestos en un solo contexto. Dentro de los diversos aspectos a considerar en el diseño esta: determinar la posición de los actuadores y los sensores, ya que estos afectan directamente la cinemática del prototipo, es fundamental ubicar el centro de rotación del robot, el lector de este documento aprenderá como afectan las dimensiones de la plataforma a la cinemática y por consiguiente el diseño electrónico y programación.

En la elaboración de tarjetas PCBs, se encuentra gran dificultad al desconocer el proceso de diseño, dibujo y construcción de la baquelita, se muestra en detalle como se hace un impreso diseñado a medida, a nivel académico; le permite a los estudiantes de electrónica fabricar sus propios diseños a partir de la información que pueden obtener de este proyecto.

Durante el diseño de la estructura mecánica es importante realizar los dibujos antes de construir las piezas para no incurrir en fallos. Se debe conocer un procedimiento de elaboración de las piezas o componentes de acuerdo a los materiales a usar en el prototipo, en el documento se evidencia el porque de proyectar antes de construir.

En el diseño del circuito electrónico, se debe tener en cuenta los componentes con las especificaciones necesarias requeridas. El desconocer estas especificaciones puede acarrear sobredimensionar, sub dimensionar y desperdiciar componentes electrónicos. Se hace necesario tener buen manejo de CAD y simuladores, los cuales facilitan la construcción de tales circuitos.

En la construcción del programa de control, se debe manejar teorías de algoritmos y procedimientos del lenguaje de programación, como también conocer el software para el desarrollo.

Con base en lo descrito anteriormente se creo el siguiente escenario de trabajo para el prototipo robot, y así lograr poner en contesto los conceptos necesarios para el diseño del mismo.

Escenario de trabajo para el robot

En una pequeña fábrica, en donde se requieren unir piezas de un producto con tornillos de diferentes longitudes y medidas, es necesario el movimiento de materiales, enseres y repuestos; tarea que en su mayoría realiza el ser humano con la ayuda de herramientas de transporte, por ejemplo carros o montacargas, y su uso depende de la cantidad requerida y las características físicas de cada uno de los objetos a transportar.

Para este proyecto se toma como referencia los tornillos almacenados en una bodega, el almacenista los lleva a cada estación según sea el pedido, por ejemplo: se recibe una llamada de la estación 5, donde se piden 70 tornillos de ¼ de pulgada rosca ordinaria; el almacenista toma los tornillos del estante y los cuenta depositándolos en una caja, luego los lleva a dicha estación caminando por el pasillo siguiendo el recorrido de la bodega a la estación 5, después de entregar el pedido regresa a la bodega sin devolverse, ver la figura 1.

Una máquina que realice dicha actividad de manera autónoma, es sin lugar a dudas una innovación para tener en cuenta, pues viene a dar soluciones a problemas cotidianos propios de la actividad.

Otros escenarios del mismo entorno:

- En ocasiones el almacenista acostumbra dejar suficientes tornillos en las estaciones de trabajo, para todo un turno. Esto genera desorden, perdida de trazabilidad en los inventarios y malas prácticas de seguridad.

- Algunas veces los trabajadores paran sus actividades para llevar los tornillos que necesitan. Se genera pérdida de tiempo en el ensamble, a causa de trasladar los materiales, baja la producción, genera congestión de pedidos en el almacén y algunos tornillos se pierden.

- Cuando el almacenista lleva los materiales a cada estación, da como resultado desatención a los proveedores, no esta al tanto del inventario, y la bodega se mantiene en desorden.

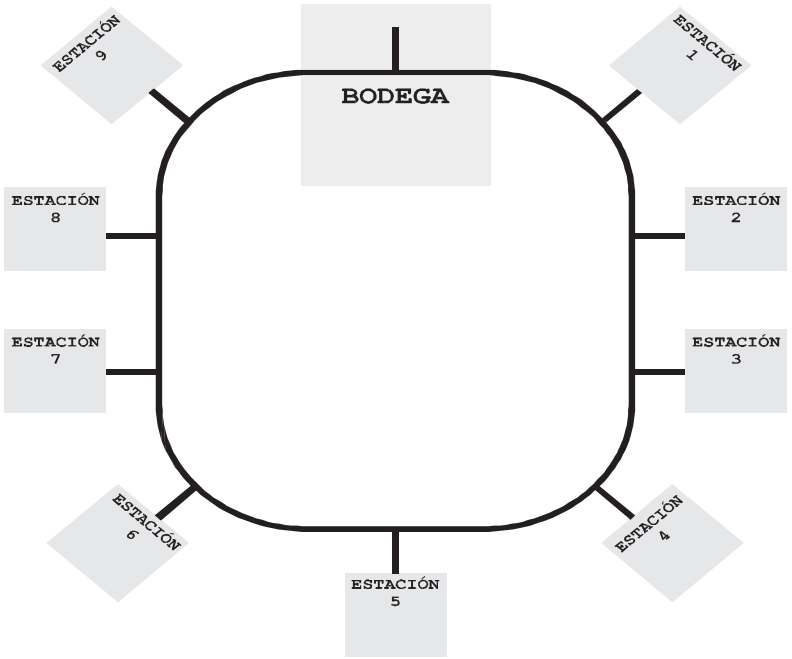


Figura 1. Esquema trazado de la línea de ensamble.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Las razones antes enunciadas son solo algunas de las innumerables que pueden ocurrir. A razón de lo descrito es necesario obtener una herramienta autónoma y programable que sirva para experimentar en el transporte de los tornillos desde la bodega hasta cada estación.

Esta herramienta debe ser sencilla de fabricar, económica y fácil de mantener.

1.2. OBJETIVOS

Objetivo general

Diseñar y construir, un robot en prototipo para transporte de tornillos.

Objetivos específicos

- Diseñar, un prototipo robot que sea guiado por seguimiento de línea, y su respectivo circuito electrónico de control.
- Construir, la plataforma mecánica para el prototipo robot y la tarjeta PCB del circuito electrónico.
- Programar, las funciones de control del prototipo robot, que lleven a realizar el recorrido dentro de la línea de ensamble, reemplazando la actividad del almacenista en el traslado de los tornillos.
- Documentar, los resultados derivados del proceso de diseño y construcción del prototipo robot.

1.3. JUSTIFICACIÓN

Se busca que este proyecto describa de forma clara y sencilla el proceso de diseño y construcción del prototipo robot, que sirva para experimentar en la actividad de transportar los tornillos (dentro del escenario supuesto a cada estación de trabajo en la línea de ensamble) y buscando como objetivo secundario evitar que los trabajadores de las estaciones y/o almacenista de la bodega, dejen sus actividades.

El documento escrito puede ser aprovechado por estudiantes, tutores, aficionados y cualquier persona con conocimientos de electrónica, como guía en la construcción de un prototipo robot, ya que ofrece detalles sobre los procedimientos de diseño y armado. Por ejemplo, el lector puede encontrar conceptos elementales de unión de componentes electrónicos con soldadura estaño, preparación de una baquelita para el proceso de grabado por degradación del cobre con producto químico, uso de herramientas de corte para trabajar el acrílico, procedimiento de unión de piezas con tornillos, preparación de piezas para obtener simetría dimensional, uso de teclado matricial, uso de pantalla LCD, uno de sensores infrarrojos, uso de sensores por ultrasonido, uso de modulo de radio frecuencia, uso de circuito electrónico puente H, uso de motores con reducción, manejo de simuladores y programas CAD para diseño de esquemas y tarjetas PCB, manejo de microcontroladores, creación de programas en asembler y en C para control, grabación de programas en el microcontrolador, técnicas de mantenimiento de un prototipo, entre otros mas procedimientos.

El prototipo tiene como característica la autonomía al realizar sus tareas, donde se le pueden asignar de manera indefinida una después de otra. Sumado a lo anterior el prototipo robot trae las siguientes ventajas para el escenario de línea de ensamble:

- Es una herramienta de transporte, que se puede usar para trasladar cualquier clase de tornillo que este dentro del parámetro de la longitud asignada.

- Trae aprovechamiento del tiempo de los trabajadores (dentro del escenario supuesto), ya que estos dedican su tiempo a sus actividades sin preocuparse por ir a traer tornillos.

- Si requiere de mantenimiento o debe recargar sus baterías, puede ser relevado por otro equipo robot del mismo modelo, sin presentar cambios en su desempeño. El controlador y el programa, del prototipo robot se pueden mejorar continuamente.

- Puede programarse, las veces que sea necesario para que realice las entregas en las estaciones.

- Su utilización, debe traer organización al proceso y el aprovechamiento del tiempo.

1.4. ALCANCES DEL PROYECTO

El proyecto al poner en contexto los aspectos de la cinemática de los robots móviles, busca que el lector adquiera las herramientas mínimas necesarias para el diseño de una plataforma funcional. La plataforma construida en este proyecto es destinada al transporte de tornillos, de longitudes máximas de 100 milímetros con diámetro de 13 milímetros y peso máximo aproximado a 20 gramos cada uno, pudiendo llevar hasta 3 kilogramos de peso. Estos componentes son de amplia aplicación, fáciles de adquirir y de diferentes tamaños, y son tomados como ejemplo, pero puede usarse cualquier objeto dentro de las características antes definidas.

Puede convertirse en la base para el diseño de futuras aplicaciones que involucren la construcción de piezas mecánicas, la manufactura de tarjetas impresas o PCBs, el manejo de CAD para el diseño de esquemas electrónicos y la construcción de sus correspondientes circuitos, la escritura de programas de control para micro controladores.

A la vez se convierte en material de consulta para los estudiantes de ingenierías electrónica, telecomunicaciones y sistemas, donde encontraran de forma sencilla y detallada manejo de programa de simulación y desarrollo para construir esquemas y circuitos electrónicos, manejo de programas compiladores de lenguaje de programación a lenguaje maquina, manejo de grabador para desarrollo de

proyectos, pueden obtener información de cómo realizar un circuito impreso con herramientas de fácil consecución en su propio taller.

El documento presenta la construcción de la plataforma o cuerpo del prototipo robot, muestra el diseño paso a paso de la tarjeta electrónica del controlador, resaltando detalles de su elaboración manual.

Se explica de forma sencilla los programas de control, tanto el de seguimiento de línea para pruebas cinemáticas, como el de control de los periféricos de interfase hombre máquina.

El prototipo robot permite determinar la calidad de los componentes electrónicos, se puede saber que reemplazar para mejorar su desempeño y permite cambiar los programas en busca de mejores resultados.

1.5. MEJORAS DEL PROYECTO

En la construcción del prototipo, se usaron componentes de fácil obtención, no obstante se limita la calidad de estos componentes, es necesario evaluar las características técnicas de cada uno y así reemplazarlos por de mejor calidad.

Debido a que la alimentación de energía es proporcionada por baterías, estas requieren ser recargadas periódicamente, se debe buscar baterías con tiempo de mayor autonomía y capacidad de corriente. Los motoreductores son de alta relación de velocidad, su desplazamiento es aproximadamente veinte centímetros por segundo, es necesario buscar motores más eficientes y veloces.

El equipo puede ser usado en experimentación y como herramienta de investigación, buscando recopilar datos que lleven a mejorar su desempeño y autonomía. De esta forma podría ser usado como herramienta de transporte en un entorno o aplicación real.

2. MARCO TEÓRICO

Los robots de la actualidad, se programan según conceptos aprendidos del comportamiento de los insectos, por ejemplo las abejas y las hormigas. Se ha encontrado como característica principal, la organización. Donde ejecutando tareas sencillas pero bien definidas, se pueden obtener procesos bastante complejos, por ejemplo las líneas de montaje, Téllez Barrera (2007).

Dichas líneas de montaje fueron desarrolladas a partir de la revolución industrial, y hoy inclinan su mirada a la automatización; gracias a dichos procesos industriales y a los trabajos hostiles se desarrolló la construcción de este tipo de máquinas. Es gracias a la computación y la inteligencia artificial, que nacieron los robots modernos.

Hoy en día hay muchos fabricantes de robots para diferentes aplicaciones; ya que llevan más de cincuenta años presentes en los procesos productivos. Países industrializados, han dedicado tiempo y trabajo en su desarrollo. Existen robots de uso doméstico, para ayuda médica, para labores peligrosas, los industriales y los llamados androides, Téllez Barrera (2007).

¿Pero que es un robot? Existen muchas maneras de definir un robot, pero la más acertada puede ser: *"Es un manipulador reprogramable y multifuncional diseñado para mover material, partes o herramientas a través de movimientos programados*" Téllez Barrera (2007).

Entonces los robots, hacen tareas sofisticadas pero no complejas; esto da por sentado un precedente y concepto inicial, "la simplicidad de sus funciones". Lo anterior es importante ya que define la manera adecuada de robot que se desea construir, y el fin, para que es concebido. Además enmarca su desarrollo óptimo y como consecuencia la función para lo que fue diseñado.

Recordando lo dicho anteriormente: un robot debe ser simple en sus funciones, de esta manera será confiable en su desempeño; debe tener una tarea definida que le permita ser óptimo; debe reprogramarse para buscar un mejor funcionamiento. El prototipo que en adelante se describirá busca cumplir con estas características.

2.1. ESTRUCTURAS BÁSICAS.

Una estructura, es la parte física mecánica de una máquina, las más usadas en robótica móvil son:

Diferencial. Esta estructura se basa en dos ruedas en un eje común, cada una controlada independientemente; donde se pueden realizar movimientos en línea recta, en arco y sobre su propio eje, Téllez Barrera (2007). Requiere mínimo de una rueda adicional para el balance de estabilidad; es bastante sencilla mecánicamente y su cinemática o manera en la que se mueve es muy sencilla, figura 2. Para lograr movimiento en línea recta, requiere de dos ruedas de tracción y que las mismas giren a la misma vez con la misma velocidad.

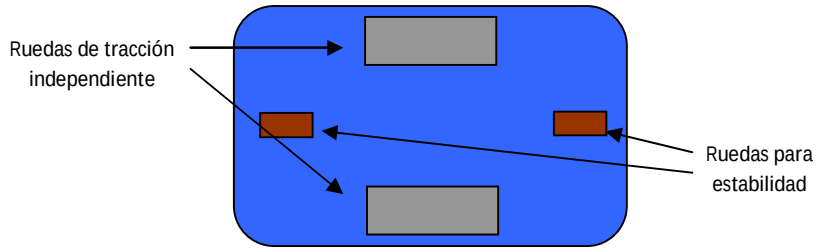


Figura 2. Estructura diferencial.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Síncrono. En esta estructura, todas las ruedas giran al mismo tiempo para el avance y vuelta. Las ruedas siempre apuntan en la misma dirección; para dar la vuelta gira sobre su propio eje de apoyo, manteniendo la posición del frente del robot, Téllez Barrera (2007). Evita inestabilidad y pérdida de contacto, pero es muy compleja mecánicamente, figura 3.

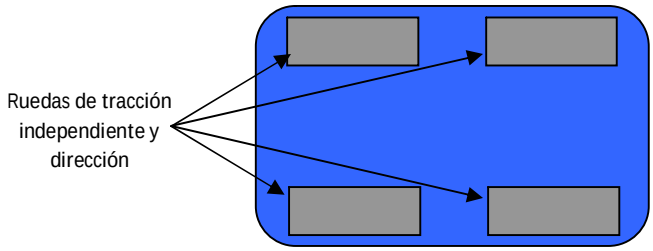


Figura 3. Estructura síncrono.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Triciclo. Esta estructura, es soportada en dos ruedas de tracción fijas, y una tercera para la dirección sin tracción; es estable, simple mecánicamente, con facilidad para los movimientos rectos. Presenta una cinemática compleja, ya que al realizar giros y vueltas requiere de grandes desplazamientos, Téllez Barrera (2007). A pesar de esta dificultad, es la seleccionada para el desarrollo de este proyecto, gracias a su facilidad de construcción, figura 4.

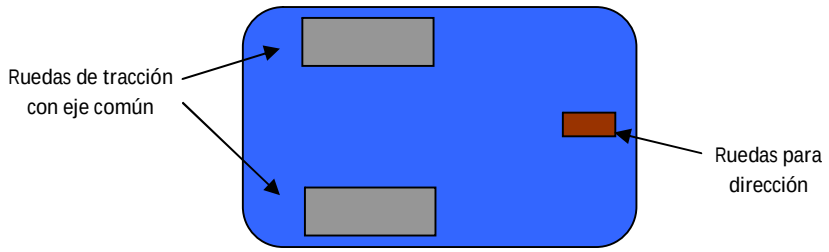


Figura 4. Estructura triciclo.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Carro. Es similar al triciclo. Cuenta con dos ruedas de tracción y dos ruedas de dirección; tiene buena estabilidad y facilidad de movimientos rectos, pero presenta mayor complejidad mecánica, debido al acoplamiento entre las ruedas de dirección, Téllez Barrera (2007). Además presenta gran complejidad cinemática, figura 5.

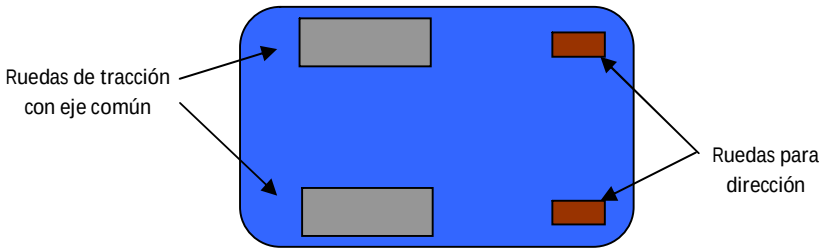


Figura 5. Estructura carro.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

2.1.1. Grados de libertad

Los grados de libertad, son la capacidad de movimientos que puede realizar un cuerpo, con respecto a un punto de referencia o posición en el espacio. Esta posición es cartesiana en X,Y, e incluye su ángulo de orientación con respecto a la posición de origen, figura 6.

En otras palabras, un grado de libertad es la capacidad de moverse a lo largo de un eje (movimiento lineal) o de rotar a lo largo del eje (movimiento rotacional). Por ejemplo, un automóvil posee 3 grados de libertad, dos de posición y uno de orientación.

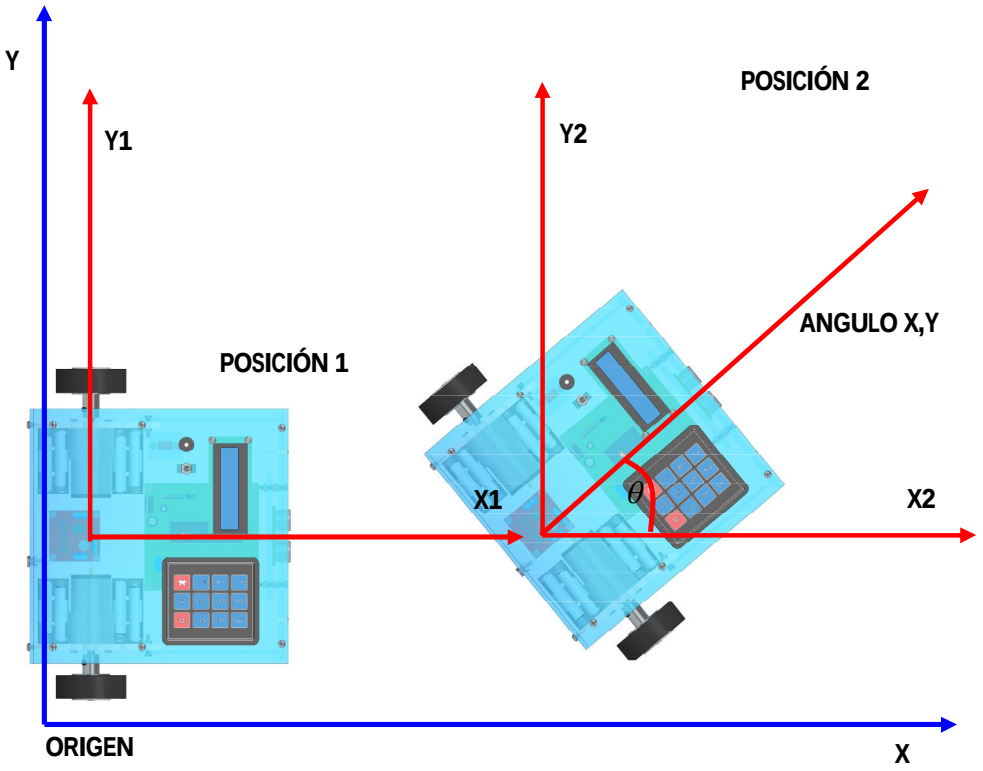


Figura 6. Grafica de cinemática de la plataforma triciclo.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Los robots móviles se encuentran cambiando de posición y de orientación con respecto a su eje de coordenadas todo el tiempo, por lo tanto deben estar en constante reconocimiento del entorno, para adaptar sus movimientos, Solaque Guzmán (2014).

El sistema de desplazamiento más básico de los robots móviles, se basa en el modelo cinemático de propulsión. De las configuraciones de robots móviles, la más utilizada, es la de tracción diferencial, gracias a que es un sistema simple y adecuado para los entornos cotidianos no muy exigentes. Este tipo de direccionamiento viene dado por la diferencia de velocidades de las ruedas, que están montadas en un único eje y son propulsadas y controladas independientemente, proporcionando tracción y dirección. Además, permite cambiar la orientación del robot sin movimientos de traslación.

Un problema importante es cómo resolver el equilibrio del robot, ya que es necesario buscar un apoyo adicional a las dos ruedas ya existentes, la solución es mediante una rueda de apoyo añadida en un diseño triangular. Otro problema que presenta la configuración de tracción diferencial, es desplazarse en línea recta, ya que para que robot pueda lograr esto, sus dos ruedas deben girar a la misma velocidad, pero si cada rueda experimenta una fricción diferente sus velocidades van a cambiar; este problema debe ser solucionado con un sistema de control dinámico, que vaya variando las velocidades a medida que el robot lo necesite. Una forma de abordar el problema, es realizar el modelo matemático que describa la cinemática del sistema, Solaque Guzmán (2014).

2.1.2. Modelado de la cinemática

Se asume que el robot se desplaza en una superficie plana idealmente sin rozamiento, se toman los ejes de las ruedas como perpendiculares al suelo por donde se desplaza. Por último el robot se debe mover únicamente por las fuerzas ejercidas por el movimiento rotacional de las ruedas, Solaque Guzmán (2014).

El robot es considerado como un mecanismo sólido, rígido y sin partes flexibles, además su desplazamiento es tomado hacia atrás y adelante, Solaque Guzmán (2014). Si se desea realizar un desplazamiento lateral, este debe ser en varios movimientos. Es necesario conocer las dimensiones del robot. Las medidas que interesan son: la distancia entre las ruedas o L y el radio de las mismas o R.

Para poder lograr un movimiento controlado, se deben tener dominio sobre la velocidad de la rueda derecha o V_r y la velocidad de la rueda izquierda o V_i.

Modelar el robot es buscar una relación directa de cómo afectan las entradas V_r y V_i, a los estados del sistema X, Y, θ. Se deben determinar los movimientos que presenta el robot, en este caso con una velocidad lineal o v y rotar con una velocidad angular ω, como se puede observar en la figura 6.

Para que el robot avance en línea recta la velocidad de sus ruedas debe ser igual, por lo tanto, se puede definir la velocidad lineal del robot, como el promedio de las velocidades de las ruedas, siendo proporcional al radio de las mismas, expresado por la ecuación:

$$v = R\frac{V_r + V_i}{2}$$

Para que el robot tenga un movimiento de rotación sobre su mismo centro, las velocidades de sus ruedas deben tener la misma magnitud pero con signo diferente, se puede definir la velocidad angular como la diferencia de la velocidad de sus ruedas sobre la longitud que hay entre ellas. Este movimiento al igual que la velocidad lineal es proporcional al radio de las ruedas, expresado por la ecuación:

$$w = R\frac{V_r - V_i}{L}$$

Luego se desarrollan las ecuaciones que definen la dinámica del movimiento de un robot en cada eje.

$$\dot{x} = v * \cos\theta$$

$$\dot{y} = v * sen\theta$$

$$\dot{\theta} = w$$

Luego se reemplazan las velocidades lineales y angulares. Con esto se obtiene el modelo del prototipo robot con configuración diferencial, Solaque Guzmán (2014).

2.2. SISTEMA ELÉCTRICO DE ALIMENTACIÓN

La alimentación eléctrica, es proporcionada por varios sistemas generadores. El más usado para aplicaciones de robótica móvil son las baterías, que proporcionan tensión eléctrica para que el robot funcione de manera autónoma, la figura 7 muestra las baterías de tipo AAA.

Figura 7. Fotografía de las baterías AAA.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Estos dispositivos, están hechos de celdas electroquímicas, que pueden convertir la energía química almacenada en electricidad. Cada celda tiene un electrodo positivo y un electrodo negativo, facilitando la salida de la energía para llevar a cabo su tarea, Bustamante (1991).

2.2.1. Pilas de Níquel Cadmio

Son conocidas como Nicads, para este caso el electrodo positivo es de óxido de níquel (NiO) y el electrodo negativo es de cadmio (Cd), y el electrolito es el óxido de potasio (KOH) que es un material altamente cáustico, Bustamante (1991). .

Existen muchas técnicas de fabricación, pero por lo general los electrodos negativos y positivos son las de las placas de níquel y cadmio saturados del hidróxido de potasio y separados por el material aislante, nylon o polipropileno.

Las placas y los separadores, forman un sándwich dando lugar a grandes celdas de almacenamiento; tiene además una ventana de seguridad, suministrada para

prevenir en una posible ruptura de la celda, debido al esfuerzo paulatino de presión, por las cargas o descargas de la pila.

La operación de carga de una celda, debe hacerse con una corriente que sea por lo menos la décima parte de la capacidad de la corriente máxima en amperios hora de la celda. Por ejemplo una pila AAA tiene una capacidad de 600 mAh o miliamperios hora; la corriente de carga debe ser de 60 mAh; se podría pensar que la pila se carga en 10 horas. A pesar de lo anterior, los fabricantes recomiendan someter la pila a un tiempo mayor a 14 horas y a carga lenta, es decir a un cuarenta por ciento más de la nominal, Bustamante (1991). .

La carga de una celda, consiste en aplicarle energía eléctrica desde una fuente de corriente directa. Al hacerlo, por el fenómeno de la electrólisis, el ácido se descompone en iones, que se polarizan formando la diferencia de potencial. Al conectar la batería a un receptor de carga, esta se va descargando lentamente, dependiendo de la capacidad en amperios hora.

La capacidad que tiene una batería, de entregar energía durante la descarga y recibirla durante la carga, se expresa en amperios hora (Ah) y esto determina su tamaño.

Por ejemplo, ¿cuánto tiempo le toma a una batería de un automóvil de 45 Ah en descargarse, si la corriente de la carga es de 5 amperios?

$$h = \frac{Ah}{A} = \frac{45}{5} = 9horas$$

Ahora esta misma batería, conectada a una carga que adsorbe 0,5 amperios ¿cuánto dura?

$$h = \frac{Ah}{A} = \frac{45}{0{,}5} = 90horas$$

Estos mismos planteamientos son usados para la carga de la batería, al conectarla a la fuente de alimentación DC, pero se debe tener en cuenta, no someterla a carga rápida, porque esta se descargará bastante rápido.

2.2.2. Acondicionamiento del voltaje

Cuando se requiere obtener voltajes exactos, los reguladores integrados de tres terminales son los más sencillos de usar. Ellos se caracterizan por reducir el rizado de entrada en 10000 veces, haciéndolo inapreciable a la salida, y por mantener un voltaje más o menos constante, Bustamante (1991). .

Algunas características básicas de estos reguladores son:

- La tensión entre los terminales Vout y GND es de un valor fijo no variable y depende del modelo que se utilice.
- La corriente que entra o sale por el terminal GND es prácticamente nula, solo se usa como referencia para el regulador.
- La tensión de entrada Vin deberá ser siempre unos 2 a 3 voltios superior a la que se desea obtener a la salida Vout para asegurar su correcto funcionamiento. Por ejemplo, en una aplicación donde se desea obtener 5 voltios en la salida, se debe tener aproximadamente de 7 a 8 voltios de entrada.

El diseño de fuentes de alimentación estabilizadas, mediante reguladores integrados monolíticos o reguladores fijos, resulta ser sumamente fácil. Concretamente para un amperio de salida, en el comercio se encuentra un encapsulado TO-220 con los siguientes valores de salida en voltaje:

- **UA7805 5 voltios**
- UA7806 6 voltios
- UA7808 8 voltios
- UA7809 9 voltios
- UA7812 12 voltios
- UA7815 15 voltios
- UA7818 18 voltios
- UA7824 24 voltios
- UA7830 30 voltios
- UA79XX voltios negativos

Todos los reguladores nombrados, tienen en común que son fijos y que proporcionan adecuadamente refrigerados, una corriente máxima de 1 amperio. Cuando un regulador está trabajando se calienta, debido a que parte de la potencia es disipada en calor. Esta potencia depende de la corriente que se entrega a la carga y de la caída de tensión que haya en el regulador, Bustamante (1991). .

2.3. ELEMENTOS DE ENTRADA

Los elementos de entrada, son los componentes electrónicos usados para introducir datos a un sistema de control. Algunos de estos componentes son los sensores, que detectan la variación física del elemento que controla, y lo transmite en forma de magnitud eléctrica. Básicamente el sensor dispone de un transductor y de un circuito o sistema amplificador de la señal. La señal que entrega un sensor debe ser recogida y en caso necesario será ampliada y acondicionada para su posterior uso.

Para el diseño de sensores, se involucran elementos eléctricos y electrónicos; estos pueden ser interruptores, elementos infrarrojos, fotorresistencias, entre otros dispositivos.

Valores de salida de los sensores

Los sensores, ayudan a trasladar los atributos del mundo físico en valores que el controlador de un robot puede usar. En general, la mayoría de los sensores pueden ser divididos en dos grandes grupos, sensores analógicos y sensores digitales.

Un sensor analógico, es aquel que puede entregar una salida variable dentro de un determinado rango. Un sensor analógico, como por ejemplo una fotorresistencia o componente que mide la intensidad de luz, puede ser cableado en un circuito que pueda interpretar sus variaciones y entregar una salida variable con valores entre 0 y 5 voltios.

Un sensor digital, es aquel que entrega una salida del tipo discreta o mejor dicho discontinua en el tiempo. En este tipo de sensores se obtiene en la salida, variación dentro de un determinado rango de valores, pero a diferencia de los sensores analógicos, estas señales van en pequeños pasos preestablecidos. Ejemplo de esta clase de sensores, son los interruptores de fin de curso.

2.3.1. Sensor infrarrojo

Es un dispositivo capaz de medir, la radiación electromagnética infrarroja de los cuerpos en su campo de visión; esta es invisible para nuestros ojos, ya que se encuentran en el espectro por debajo de la luz visible. En la figura 8, se muestran tres sensores QTR-1A, de salida análoga tipo reflex en board, diseñado para aplicaciones de detección de contraste y seguimiento de línea, muy usados para detectar los diferentes contrastes de colores, Palacios (2008).

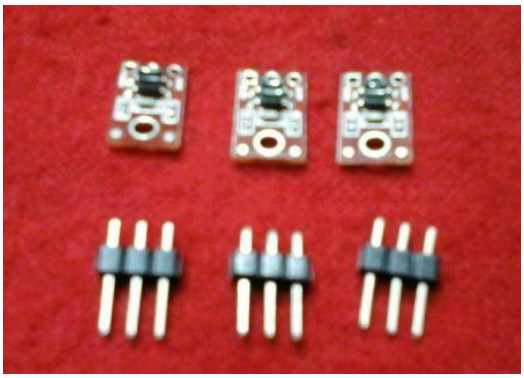


Figura 8. Fotografía sensores reflectivos.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Formados por un fototransistor y un LED infrarrojo; los rayos infrarrojos son emitidos por el IRLED, este debe conectarse siempre respetando su polaridad, de lo contrario no se ilumina. Dado que el IRLED es muy pequeño, se señalan el ánodo y el cátodo por la longitud de las patas. La pata larga corresponde al ánodo que se conecta el polo (+) y la pata corta corresponde al cátodo al que se conecta el polo (-).

Estos diodos, se diferencian de los LED normales por el color de la cápsula que los envuelve, generalmente azul o gris. El diámetro ésta alrededor de 5 mm. Los rayos infrarrojos se caracterizan por ser portadores de calor radiante, estos mismos son producidos en mayor o menor intensidad por cualquier objeto a temperatura superior al cero absoluto.

El fototransistor es un fotodetector que trabaja como un transistor clásico, pero normalmente no tiene conexión base. En estos transistores la base está reemplazada por un cristal fotosensible que cuando recibe luz, produce una corriente y desbloquea el transistor. En el fototransistor la corriente circula sólo en un sentido y el bloqueo del transistor depende de la luz; cuanta más luz hay más conduce, Palacios (2008).

El principio del fototransistor, es aparentemente el mismo que el del transistor clásico. Pero si observamos el componente se ve que sólo posee dos pines, un emisor y un colector, pero le falta la base. La base de hecho es sustituida por una capa de silicio fotosensible. Si esta capa está iluminada aparece en la base una corriente que crece con la luz, lo que pone en marcha al transistor. El fototransistor reacciona con la luz visible y con los rayos infrarrojos que son invisibles.

Para distinguirlo del IRLED su cápsula es transparente. En el fototransistor, al igual que en los IRLED, la polaridad viene dada por la longitud de sus patas pero con una diferencia muy importante; en el fototransistor la pata larga es el negativo (-), al revés que en los LED, que es el positivo (+), Palacios (2008).

Sensores reflexivos

Este tipo de sensor, presenta una cara frontal en la que encontramos tanto al LED como al fototransistor, la figura 9 representa su aspecto físico visto de frente.

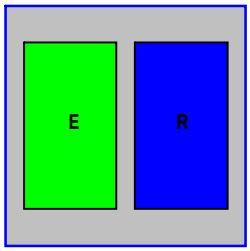


Figura 9. Esquema del sensor reflectivo.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Debido a esta configuración el sistema tiene que medir la radiación proveniente del reflejo de la luz emitida por el LED. Se tiene que tener presente que esta configuración es sensible al ambiente, perjudicando las medidas, pueden dar lugar a errores; es necesario la incorporación de circuitos de filtrado en términos de longitud de onda, así pues será importante que trabajen en ambientes de luz controlada. Otro aspecto a tener en cuenta es el coeficiente de reflectividad del objeto, el funcionamiento del sensor será diferente según el tipo de superficie.

Están diseñados especialmente para la detección, clasificación y posicionado de objetos; la detección de formas, colores y diferencias de superficie, incluso bajo condiciones ambientales extremas.

2.3.2. Sensor por Ultrasonido

Básicamente, los ultrasonidos son eso mismo, sonido; exactamente igual que los que oímos normalmente, salvo que tienen una frecuencia mayor que la máxima audible por el oído humano, que inicia desde unos 16 Hz y tiene un límite superior de aproximadamente 20 KHz, mientras que el ultrasonido tiene una frecuencia de 40 KHz, Téllez Barrera (2007).

El funcionamiento básico de los ultrasonidos como medidores de distancia, se muestra de una manera clara en la figura 10, donde se tiene un transmisor que emite un pulso de ultrasonido, dicho pulso viaja y al encontrar objeto rebota. La reflexión es detectada por el receptor de ultrasonidos y se tiene en cuenta el tiempo que se gasta, entre el pulso inicial y la llegada de la señal.

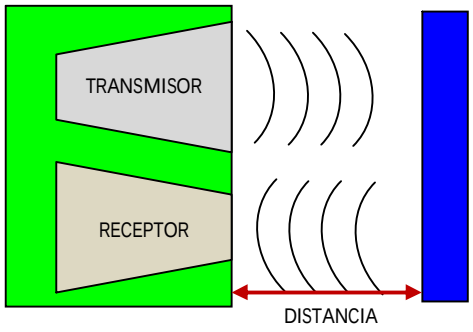


Figura 10. Esquema funcionamiento básico.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La mayoría de los sensores de ultrasonido de bajo costo, se basan en la emisión de un pulso de ultrasonido cuyo campo de acción, es de forma cónica, Téllez Barrera (2007). Midiendo el tiempo que transcurre entre la emisión del sonido y la percepción del eco se puede establecer la distancia a la que se encuentra el obstáculo que ha producido la reflexión de la onda sonora, mediante la fórmula:

$$d = \frac{1}{2} V * t$$

Donde V es la velocidad del sonido en el aire y t es el tiempo transcurrido entre la emisión y recepción del pulso.

El eco que se recibe como respuesta a la reflexión del sonido, indica la presencia del objeto más cercano que se encuentra dentro del cono acústico y no especifica en ningún momento la localización angular del mismo. Aunque la máxima probabilidad, es que el objeto detectado esté sobre el eje central del cono acústico, la probabilidad de que el eco se haya producido por un objeto presente en la periferia del eje central no es en absoluto despreciable y ha de ser tenida en cuenta y tratada convenientemente, Téllez Barrera (2007).

La cantidad de energía acústica reflejada por el obstáculo, depende en gran medida de la estructura de su superficie. Para obtener una reflexión altamente difusa del obstáculo, el tamaño de las irregularidades sobre la superficie reflectora debe ser comparable a la longitud de onda de la onda de ultrasonido incidente.

En los sensores de ultrasonido, de bajo costo se utiliza el mismo transductor como emisor y receptor. Tras la emisión del ultrasonido se espera un determinado tiempo a que las vibraciones en el sensor desaparezcan y esté preparado para recibir el eco producido por el obstáculo. Esto implica que existe una distancia mínima d, a partir de la cual el sensor mide con precisión. Por lo general, todos los objetos que se encuentren por debajo de esta distancia, d, serán interpretados por el sistema como que están a una distancia igual a la distancia mínima.

Los factores ambientales tienen una gran repercusión sobre las medidas:

Las ondas de ultrasonido se mueven por un medio material que es el aire. La densidad del aire depende de la temperatura, influyendo este factor sobre la velocidad de propagación de la onda, García Breijo (2008), según la ecuación:

$$V_S = V_{SO}\sqrt{1+\frac{T}{273}}$$

Donde V_{SO} la velocidad de propagación de la onda sonora a 0ºC, y T la temperatura absoluta (grados Kelvin). La temperatura afecta a la capacidad de detección, García Breijo (2008).

Un factor de error muy común, es el conocido como falsos ecos. Puede darse el caso en que la onda emitida por el transductor se refleje varias veces en diversas superficies antes de que vuelva a incidir en el transductor. Este fenómeno, conocido como reflexiones múltiples, implica que la lectura del sensor evidencia la

presencia de un obstáculo a una distancia proporcional al tiempo transcurrido en el viaje de la onda; es decir, una distancia mucho mayor que a la que está en realidad el obstáculo más cercano, que pudo producir la primera reflexión de la onda. Otra fuente más común de falsos ecos, conocida como crosstalk, se produce cuando se emplea un cinturón de ultrasonidos donde una serie de sensores están trabajando al mismo tiempo. En este caso puede ocurrir que un sensor emita un pulso y sea recibido por otro sensor que estuviese esperando el eco del pulso que él había enviado con anterioridad, García Breijo (2008).

Las ondas de ultrasonido, obedecen a las leyes de reflexión de las ondas, por lo que una onda de ultrasonido tiene el mismo ángulo de incidencia y reflexión respecto a la normal a la superficie. Esto implica que si la orientación relativa de la superficie reflectora con respecto al eje del sensor de ultrasonido es mayor que un cierto umbral, el sensor nunca recibirá el pulso de sonido que emitió, García Breijo (2008).

Modulo ultrasonido SRF04

Utiliza dos traductores de ultrasonidos, uno emisor y otro receptor, basándonos en el tiempo que tarda la señal en ir desde el emisor hasta el objeto obstáculo y volver rebotada desde éste hasta el receptor. Midiendo dicho tiempo podemos calcular con suficiente precisión y exactitud la distancia entre el objeto y nuestros traductores, Palacios (2008).

Las características del modelo SRF04 de la casa Británica Robot Electronics son:

- Tensión 5V
- Consumo 30 mA Típico 50mA Max.
- Frecuencia: 40 Khz.
- Distancia Mínima: 3 cm.
- Distancia Máxima: 300 cm.
- Sensibilidad: Detecta un palo de escoba a 3 m.
- Pulso de Disparo 10 uS min. TTL
- Pulso de Eco: 100 uS - 18 mS
- Retardo entre pulsos: 10 mS Mínimo
- Pulso de Eco: 100 uS - 18 mS
- Tamaño: 43 x 20 x 17 mm
- Peso: 10 gr.

Este sensor tiene la particularidad de manejarse solo con dos hilos, aparte de los de alimentación. Por uno de ellos se le envía el pulso de disparo o trigger y por el otro recibimos el pulso de eco o echo, cuya amplitud es directamente proporcional a la distancia a la que ha sido detectado el obstáculo interpuesto, figura 11 muestra el aspecto físico del sensor, Palacios (2008)..

Figura 11. Fotografía sensor de ultrasonido.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Para implementar de sensor se deben tener en cuenta los siguientes pasos:

- Emitir un pulso ultrasónico de corta duración.
- Esperar el inicio del pulso de eco resultante y poner a cero un temporizador.
- Esperar el fin del pulso de eco y guardar el estado de dicho temporizador al ocurrir esto.
- Calcular el tiempo de duración de dicho eco, que como sabemos es proporcional al tiempo empleado en ir y venir desde el objeto.
- Calcular la distancia sabiendo el tiempo empleado en recibir el eco y la velocidad de la onda reflejada.

2.3.3. Teclado matricial

El teclado matricial, proporciona una interfaz sencilla de entrada de datos, Palacios (2008). Sus usos pueden ser tan variados como aplicaciones que precisen de la introducción manual, la figura 12 es una foto donde se muestra se forma física.

Esencialmente están constituidos por filas y columnas conductoras, en cuyo cruce se encuentra un pulsador mecánico o de membrana, que al ser pulsado establece el contacto eléctrico entre la fila y la columna correspondiente.

.

Figura 12. Fotografía del teclado matricial.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Al observar el dorso de un teclado típico, se ven los pines donde cada uno de ellos corresponde o bien a una fila de las cuatro o a una columna de las tres, el teclado matricial 4 x 3 está formado por 12 pulsadores SPDT, organizados en 4 filas y 3 columnas, ideal para ser utilizados en sistemas basados en microcontroladores. Por ejemplo al presionar la tecla “1”, estaremos conectando la columna 1 con la fila 1; si pulsamos la tecla “4”, se esta conectando nuevamente la columna 1, esta vez con la fila 2; si pulsamos la tecla “9”, se ha conectando la columna 3 con la fila 3.

Ahora bien, si no se dispone del datashet del teclado, se puede encontrar la relación de los pines con las filas/columnas utilizando la funcionalidad de medir continuidad con el multímetro. Se conecta el pin 1 y el pin 2 al multímetro e ir pulsando los diferentes botones, hasta encontrar continuidad, luego se pasa a conectar el pin 1 y pin 3 repitiendo el proceso. Apuntamos los datos, según las pruebas y podemos determinar la configuración de interruptores para cada tecla, Palacios (2008).

2.3.4. Modulo RF M40 RFC

Si se quiere tener control remoto sencillo, en la elaboración de proyectos, se puede usar un dispositivo, que consta de un transmisor y receptor para el envío y recepción de órdenes. Hoy en día disponemos a un costo razonable módulos de RF, transmisor y receptor, diseñados para ponerlos a funcionar con pocos componentes adicionales.

El circuito de referencia **M40 RFC PT2262/PT2272 4-CH**, que se alimenta con tensión de 12 voltios en el emisor, a una corriente de 10 miliamperios y una potencia de radiación de 10mw. Trabaja basado en el sistema de modulación ASK a una frecuencia de transmisión de 315MHz que cubre una distancia de transmisión de 50 a 100 metros, una sensibilidad de receptor de 100dBm, Palacios (2008).

La salida es codificada a 5 voltios, nivel alto corresponde a la mitad del voltaje de alimentación de la tarjeta receptora y el nivel bajo es de 0,7 voltios, con sensibilidad de -98 decibeles. Tiene siete pines llamados: **VT, D3, D2, D1, D0, +5 V, GND**. El emisor tiene 4 botones los cuales controlan las salidas D0, D1, D2, D3 en el módulo de recepción.

Concepto de dBm

El dBm es una unidad de medida utilizada principalmente en telecomunicación, para expresar la potencia absoluta mediante una relación logarítmica. El dBm se define como el nivel de potencia en decibelios en relación a un nivel de referencia de 1 mW. El valor en dBm en un punto, donde tenemos una potencia P, viene dado por la fórmula siguiente:

$$dBm = 10X \log \frac{P}{1mW}$$

Al utilizarse un nivel de referencia determinado (1 mW) la medida en dBm constituye una verdadera medición de la potencia y no una simple relación de potencias como en el caso de la medida en decibelios. Así, una lectura de 20 dBm significa que la potencia medida es 100 veces mayor que 1mW y por tanto igual a 100 mW. Puesto que se trata de una relación logarítmica, podemos cometer error al comparar por simple inspección potencias medidas en dBm, Palacios (2008).

2.4. ELEMENTO CONTROLADOR

El sistema de control, es el encargado de procesar los datos de información provenientes de los sensores y según su programación da respuesta y maneja los actuadores del robot, Téllez Barrera (2007). Para tal tarea es fiable el uso de un microcontrolador, a causa de algunas ventajas que se enuncian a continuación:

Un microcontrolador es un circuito integrado, que contiene todos los componentes de un computador. Muy usados para controlar el funcionamiento de una tarea determinada y debido a su reducido tamaño, suele ir incorporado en el propio dispositivo al que gobierna. Esta última característica es la que le confiere la denominación de controlador incrustado, conocido como computador dedicado. En su memoria sólo reside un programa destinado a gobernar una aplicación determinada; sus líneas de entrada o salida soportan la conexión de sensores y actuadores del dispositivo a controlar. Una vez programado y configurado el microcontrolador solamente sirve para gobernar la tarea asignada.

2.4.1. Microcontrolador PIC16F877A

Hasta antes de 1971, la mayor parte de las aplicaciones digitales en electrónica se basaban en la llamada lógica cableada. Es decir, si existía un problema este era analizado y se sintetizaba una función en base a la lógica de boole, dando la solución al problema planteado. Con los microprocesadores, se varió el esquema de diseño de tal forma que un problema era descompuesto en una serie de tareas más simples; el microprocesador ejecutaba una serie de pasos o instrucciones para llevar a efecto cada una de las tareas, en ocasiones no era necesario volver a armar un circuito para solucionar otro problema, sino que se cambiaba las instrucciones o programa para obtener otra aplicación. Se especializan en aplicaciones donde resuelven problemas, como por ejemplo: en los teclados o mouse de las computadoras, en los electrodomésticos, en la industria automotriz, en el procesamiento de imagen y video, Palacios (2008).

Resumiendo, el microcontrolador es un circuito integrado programable que contiene todos los componentes necesarios para controlar el funcionamiento de una tarea determinada, utiliza muy pocos componentes asociados. Las ventajas de trabajar este tipo de componentes son: su bajo precio, reducido consumo, tamaño pequeño, gran calidad, fiabilidad y gran cantidad de información.

La figura 13, muestra el diagrama de pines del microcontrolador PIC16F877A fabricado por **Microchip Technology Inc**, esta corresponde a una vista de la forma física del integrado visto por encima.

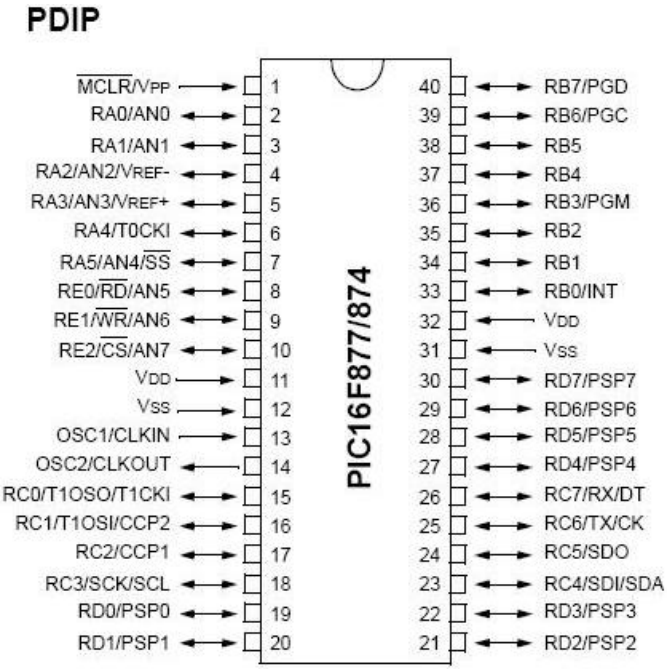


Figura 13. Diagrama de pines microcontrolador.

Fuente: Hoja de datos proporcionada por Microchip Technology Inc © 2007.

Algunas características

Puede trabajar con una frecuencia máxima de 20 MHz, normalmente se alimenta con 5 voltios, aplicados en los pines V_{DD} y V_{SS}, que son la alimentación y la masa del integrado. El consumo de corriente para el funcionamiento del microcontrolador depende de la tensión de alimentación, de la frecuencia de trabajo y de las cargas que soporta, siendo del orden de los miliamperios. El circuito de alimentación del microcontrolador debe tratarse como cualquier otro circuito digital, debiendo conectarse un condensador de desacoplo de unos 100 nano faradios, lo más cerca posible de los pines de alimentación, data sheet (2001).

Este microcontrolador contiene en su interior 8k palabras de memoria de programa tipo flag, 368 bytes de memoria RAM y 256 bytes de memoria EEPROM. Una de las grandes ventajas es su memoria flash y EEPROM, en la que los programas se pueden grabar y borrar eléctricamente; además se puede reprogramar de manera muy fácil.

El microcontrolador se comunica con el mundo exterior a través de los puertos, estos están constituidos por líneas digitales de entrada/salida que trabajan entre 0 y 5 voltios. Los puertos se pueden configurar como entradas para percibir los datos o como salidas para gobernar dispositivos externos. Las líneas de los puertos se pueden programar individualmente como entradas o como salidas y se usan de la misma forma, data sheet (2001).

Al conectar por primera vez el microcontrolador el bit **RP0** del registro **STATUS** se carga automáticamente con cero, dando paso al acceso del banco cero de la memoria de datos. Para configurar las líneas de los puertos hay que ingresar registro **STATUS,** en el bit RP0 cambiar su contenido por uno y luego se cargan los registros **TRISA** y **TRISB** con los datos deseados.

Las líneas de los puertos entregan niveles TTL, cuando la tensión de alimentación aplicada al microcontrolador es de 5 voltios. Cada línea puede suministrar una corriente máxima de 20 miliamperios, cuando está en nivel alto y pueden absorber una corriente máxima de 25 miliamperios, cuando está en un nivel bajo. La limitación de disipación máxima de potencia del integrado para el puerto A, es de 50 miliamperios en nivel alto y 80 miliamperios en nivel bajo y para el puerto B, es de 100 miliamperios en nivel alto y 150 miliamperios en nivel bajo.

Las principales características con que cuenta el microprocesador PIC16F877A son:

- Procesador de arquitectura RISC avanzada.
- Juego de 35 instrucciones con 14 bits de longitud.
- Frecuencia de 20 Mhz.
- Hasta 8K palabras de 14 bits para la memoria de código, tipo flash.
- Hasta 368 bytes de memoria de datos RAM.
- Hasta 256 bytes de memoria de datos EEPROM.
- Hasta 14 fuentes de interrupción internas y externas.
- Pila con 8 niveles.
- Modos de direccionamiento directo, indirecto y relativo.
- Perro guardián (WDT).
- Código de protección programable.
- Modo Sleep de bajo consumo.
- Programación serie en circuito, con 2 pines.
- Voltaje de alimentación, comprendido entre 2 y 5.5 voltios.
- Bajo consumo, menos de 2 mA a 5 V y 5 Mhz.

2.4.2. Conexión básica del microcontrolador

Los microcontroladores son muy fáciles de trabajar a nivel de hardware, gracias a que tienen muy pocas conexiones a componentes externos; es sencillo en el manejo y contiene un buen promedio en los parámetros como: velocidad, consumo, tamaño, alimentación. Por esto es uno de los componentes electrónicos más utilizados, la figura 14 muestra la forma física del circuito integrado.

Figura 14. Fotografía circuito integrado PIC16F877A.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Su conexionado es bastante sencillo. Iniciando con el pin 1 MCLR, debe ser conectado a 5 voltios de forma continua por medio de una resistencia de 100 ohmios, en serie con una resistencia de 10 mil ohmios, pero en algunas ocasiones la resistencia de 100 ohmios se reemplaza por un diodo 1N4148, de alta velocidad de conmutación, para evitar entradas de rebotes o ruidos eléctricos. Este pin tiene como función principal habilitar un reinicio del programa de forma externa por Hardware, usando un pulsador normal abierto conectado a cero voltios o negativo.

Los pines 13 y 14, están conectados a el cristal de 4 mega Hertz y los condensadores de 22 pico faradios son para desacoplo o como filtros de ruido eléctrico.

El puerto B, es de entrada o salida digital de 8 bits, con pull up, muy usado en aplicación de teclado matricial. El pull up se puede activar limpiando o poner cero, en el bit 7 del registro OPTION_REG, Palacios (2008).

El puerto C, es de entrada o salida digital de 8 bits, que multiplexa algunas funciones para sus líneas. Las líneas multiplexadas más aplicadas son:

- RC2 que puede ser un pin de entrada o salida digital o la salida de una onda de PWM generada a partir de un recurso de HW denominado módulo CCP.
- RC6 y RC7 que puede ser pines de entrada o salida digitales o los pines de comunicación para el USART Tx y Rx respectivamente. Como los niveles de salida en estos pines son CMOS, es necesario conectarlos a un chip que convierta niveles TTL / CMOS a RS232, ejemplo MAX232.
- RC3 y RC4 son pines que pueden configurarse como entrada o salida para la comunicación I2C, para la interconexión de circuitos integrados.

El puerto D, es de entrada o salida digital de 8 bits, que multiplexa funciones con el periférico denominado puerto paralelo esclavo PSP.

El puerto E, es de entrada o salida de 3 pines RE0, RE1 y RE2, cuyas líneas pueden ser configuradas como entradas o salidas digitales o entradas análogas.

La figura 15, muestra el esquema básico, no se muestran los pines de alimentación, pero son marcados como VDD referencia a cero o negativo y VSS 5 voltios.

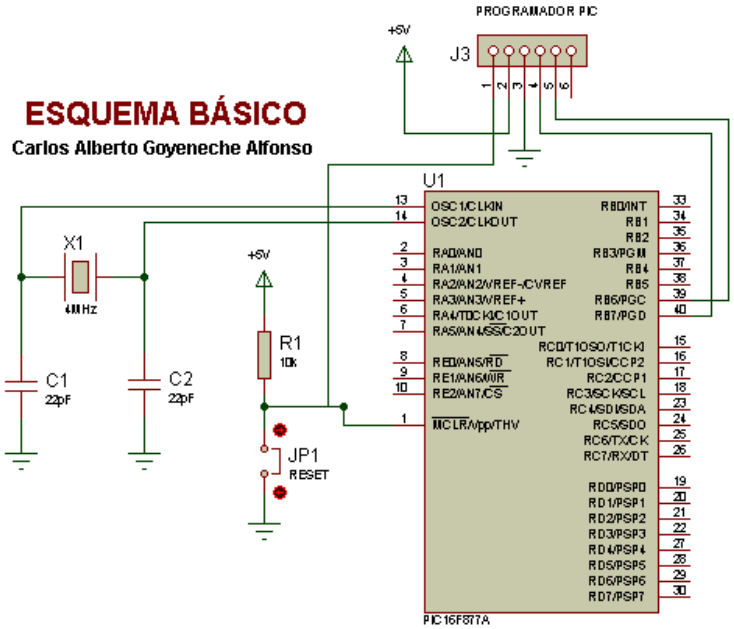


Figura 15. Esquema circuito básico del PIC.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

2.4.3. Instrucciones del microcontrolador

El microcontrolador, tiene 35 instrucciones para realizar su programación, son simples y rápidas, la mayoría de las instrucciones se ejecutan en cuatro ciclos de reloj, menos las de salto que requieren ocho ciclos de reloj, casi todas las instrucciones pueden usar cualquier operando, y todas las instrucciones tiene la misma longitud de catorce bits y los datos son de ocho bits, Palacios (2008).

- **Instrucciones orientadas a los bytes**

Nemónico	Parámetros	Descripción	Ciclos	Banderas
ADDWF	f, d	Add W and f	1	C, DC, Z
ANDWF	f, d	AND W with f	1	Z
CLRF	f	Clear f	1	Z
CLRW	-	Clear W	1	Z
COMF	f, d	Complement f	1	Z
DECF	f, d	Decrement f	1	Z
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	None
INCF	f, d	Increment f	1	Z
INCFSZ	f, d	Increment f, Skip if 0	1(2)	None
IORWF	f, d	Inclusive OR W with f	1	Z
MOVF	f, d	Move f	1	Z
MOVWF	f	Move W to f	1	None
NOP	-	No Operation	1	None
RLF	f, d	Rotate left f through carry	1	C
RRF	f, d	Rotate right f through carry	1	C
SUBWF	f, d	Subtract W from f	1	C, DC, Z
SWAPF	f, d	Swap nibbles in f	1	None
XORWF	f, d	Exclusive OR W with f	1	Z

Tabla 1. Instrucciones orientadas a los bytes.

Fuente: Hoja de datos proporcionada por Microchip Technology Inc © 2007.

- **Instrucciones orientadas a los bits**

Nemónico	Parámetros	Descripción	Ciclos	Banderas
BCF	f, b	Bit Clear f	1	None
BSF	f, b	Bit Set f	1	None
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	None
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	None

Tabla 2. Instrucciones orientadas a los bits.

Fuente: Hoja de datos proporcionada por Microchip Technology Inc © 2007.

- **Operaciones con literales y de control**

Nemónico	Parámetros	Descripción	NroCic.	Banderas
ADDLW	k	Add literal and W	1	C, DC, Z
ANDLW	k	AND literal with W	1	Z
CALL	k	Call subroutine	2	
CLRWDT	-	Clear Watchdog Timer	1	TO,PD
GOTO	k	Go to address	2	None
IORLW	k	Inclusive OR literal with W	1	Z
MOVLW	k	Move literal to W	1	None
RETFIE	-	Return from interrupt	2	None
RETLW	k	Return with literal in W	2	None
RETURN	-	Return from Subroutine	2	None
SLEEP	-	Go into standby mode	1	TO,PD
SUBLW	k	Subtract W from literal	1	C, DC, Z
XORLW	k	Exclusive OR literal with W	1	Z

Tabla 3. Instrucciones con literales.

Fuente: Hoja de datos proporcionada por Microchip Technology Inc © 2007.

Microchip recomienda no utilizar las instrucciones **TRIS** y **OPTION**, para mantener la compatibilidad con el PIC16F877A, Palacios (2008).

2.4.4. Consideraciones para la programación

El lenguaje que entiende el microcontrolador, está formado por unos y ceros del sistema binario. Cualquier instrucción esta expresada en binario denominado lenguaje de máquina, por ejemplo “11111000111010” equivale a la instrucción “suma 58 al registro de trabajo W y guarde el resultado en este mismo registro”. Dicha codificación binaria es bastante incómoda para trabajar, por lo que se usa la codificación hexadecimal, para facilitar la interpretación, García Breijo (2008).

El lenguaje de máquina es difícil de usar, ya que aleja al hombre de su forma natural de expresarse; por esto se utiliza el lenguaje ensamblador, es un poco mas cómodo de trabajar. Este lenguaje utiliza neumónicos que son grupos de caracteres alfanuméricos que simbolizan las órdenes o tareas a realizar con cada instrucción.

El programa ensamblador, es un software que se encarga de traducir los neumónicos y símbolos alfanuméricos del programa escrito en ensamblador a código máquina, para que pueda ser interpretado y ejecutado por el microcontrolador. El programa en lenguaje ensamblador recibe la denominación de código fuente y tiene la extensión **.asm**. El código fuente es traducido a máquina por el programa ensamblador, dando como resultado los ficheros para ser grabados en la memoria del microcontrolador y tiene la extensión **.hex**.

Se guardan todas las instrucciones del programa, de forma permanente en la memoria interna del microcontrolador **PIC16F877A**, donde se distinguen tres bloques:

- **Memoria de programa**. Con 8k posiciones, contiene el programa con las instrucciones que manejan las tareas. Es del tipo no volátil, esto quiere decir que el programa se mantiene aunque se retire la energía del microcontrolador.

- **Memoria de datos RAM**. Se destina para guardar las variables y los datos. Es del tipo volátil, esto quiere decir que los datos se borran cuando desaparece la energía en el microcontrolador.

- **Memoria EEPROM de datos**. Es una pequeña área de memoria de datos de lectura y escritura no volátil. Esta permite que no se pierda la información del programa.

Al realizar un proyecto, es importante conocer la programación estructurada, que se entiende como la división de un programa en módulos o procedimientos, que realizan determinada tarea y se ejecutan uno luego de otro. Un procedimiento consta de un conjunto de pasos para llevar a cabo una tarea. En el lenguaje ensamblador las subrutinas hacen esta función. Donde las subrutinas deben tener algunas características:

- Deben ser independientes. La modificación de una subrutina no debe ocasionar la redefinición de otras.
- Plantear la subrutina de forma que el programa le proporcione unos datos de entrada, esta los procesa y devuelve los resultados.
- Se debe definir claramente los registros de entrada y de salida.
- El programa principal debe estructurarse con un solo punto de entrada y un solo punto de salida, porque es más fácil de entender.
- Las instrucciones de salto deben usarse en lo posible en el programa principal.
- Las subrutinas se deben diseñar para que puedan ser utilizadas en diferentes programas sin dificultad.

Las ventajas de la programación estructurada son:

- Simplifica el desarrollo de cada parte del algoritmo por separado, permitiendo concentrar la atención en los pequeños detalles.
- Produce programas que son más fiables, fáciles de entender, documentar y modificar.

La utilización de llamada a subrutina con CALL, en lugar de saltos como GOTO aprovecha las ventajas de la programación estructurada, pero se corre el riesgo de desbordar la pila, hay que tener especial precaución de no superar los ocho niveles de anidamiento de subrutinas, Palacios (2008).

2.5. ELEMENTOS DE SALIDA

Son componentes electrónicos, que permiten presentar resultados que originan respuesta física, tales como: movimiento producido por los motores DC; luz visible producido diodos emisores de luz; sonido producido por buzzer o zumbadores y los más elaborados como las pantallas LCD que presentan mensajes.

2.5.1. Motor de corriente continúa

Un motor de corriente continua, figura 16, posee un par de arranque elevado y su velocidad se puede regular con facilidad, lo que les hace ideales para ciertas aplicaciones eléctricas y en todas aquéllas en que sea muy importante el control y la regulación de las características funcionales del motor.

Figura 16. Fotografía del motor DC con reductor de velocidad.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Los motores eléctricos, se basan en las fuerzas que aparecen en los conductores cuando son recorridos por corrientes eléctricas y. a su vez, están sometidos a la acción de un campo magnético.

Los polos magnéticos del imán, son los encargados de producir el campo magnético inductor. La espira, que se ha situado en el rotor, es recorrida por una corriente continua que se suministra a través de un anillo de cobre cortado por la mitad. Las dos mitades se aíslan eléctricamente y se sitúa sobre ellas unos contactos deslizantes de carbón, llamados escobillas, de tal forma que la corriente aplicada por la fuente de alimentación pueda llegar a los conductores del rotor.

Como las corrientes que circulan por ambos lados de la espira son contrarias, al aplicar la regla de la mano izquierda, se comprueba que aparecen fuerzas contrarias en cada lado activo de la espira, lo que determina un par de giro. Para que el sentido de giro sea siempre el mismo, el par de fuerzas siempre deberá actuar en el mismo sentido.

En el caso de que los conductores de la espira girasen hasta enfrentarse con el polo contrario, con el mismo sentido de corriente que en la anterior posición, la fuerza se invertiría de sentido y la espira no establecería nunca una revolución. Con el colector de delgas se resuelve este problema, haciendo que la corriente siempre circule en el mismo sentido respecto al campo magnético, dando como resultado que el colector actúe como un rectificador, Rodríguez Pozueta (2015).

Componentes de un motor DC

El motor de corriente continua está compuesto de dos piezas fundamentales, el rotor y el estator.

El Rotor, constituye la parte móvil del motor, proporciona el torque para mover a la carga, Rodríguez Pozueta (2015). Está formado por:

- **Eje:** Formado por una barra de acero fresada. Imparte la rotación al núcleo, devanado y al colector.
- **Núcleo:** Se localiza sobre el eje. Fabricado con capas laminadas de acero, su función es proporcionar un trayecto magnético entre los polos para que el flujo magnético del devanado circule.
- **Devanado:** Consta de bobinas aisladas entre sí y entre el núcleo de la armadura. Estas bobinas están alojadas en las ranuras, y están conectadas eléctricamente con el colector, el cual debido a su movimiento rotatorio, proporciona un camino de conducción conmutado.
- **Colector:** Denominado conmutador, está constituido de láminas de material conductor llamadas delgas, separadas entre sí y del centro del eje por un material aislante, para evitar cortocircuito con dichos elementos. El colector se encuentra sobre uno de los extremos del eje del rotor, de modo que gira con éste y está en contacto con las escobillas. La función del colector es recoger la tensión producida por el devanado inducido, transmitiéndola al circuito por medio de las escobillas.

El Estator, constituye la parte fija de la máquina. Su función es suministrar el flujo magnético que será usado por el bobinado del rotor para realizar su movimiento giratorio, Rodríguez Pozueta (2015). Está formado por:

- **Armazón:** Denominado yugo, tiene dos funciones primordiales: servir como soporte y proporcionar una trayectoria de retorno al flujo magnético del rotor y del imán permanente, para completar el circuito magnético.

- **Imán permanente:** Compuesto de material ferromagnético altamente remanente, se encuentra fijado al armazón o carcaza del estator. Su función es proporcionar un campo magnético uniforme al devanado del rotor o armadura, de modo que interactúe con el campo formado por el bobinado, y se origine el movimiento del rotor como resultado de la interacción de estos campos.

- **Escobillas:** Las escobillas están fabricadas de carbón, y poseen una dureza menor que la del colector, para evitar que éste se desgaste rápidamente. Se encuentran albergadas por los porta escobillas. Ambos, escobillas y porta escobillas, se encuentran en una de las tapas del estator. La función de las escobillas es transmitir la tensión y corriente de la fuente de alimentación hacia el colector y, por consiguiente, al bobinado del rotor.

2.5.2. Diodo LED

Los primeros ejemplares comerciales de estos dispositivos, aparecieron en 1963, a un precio bastante elevado y con un rendimiento luminoso extremadamente bajo. En la actualidad los LED figura 17, se producen en millones de ejemplares y el precio es insignificante, Pro-Lighting (2015).

Se encuentran en el mercado con luz roja, ámbar, amarilla, verde e infrarrojo. Los tipos especiales, como los bicolores, los de alto brillo, los "flashing leds" que incluyen en su caja estándar un circuito integrado, los de luz azul y luz blanca, todos a un precio ampliamente justificado, Pro-Lighting (2015).

Figura 17. Fotografía de los diodos LED.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

El diodo emisor de luz, funciona gracias a la teoría de la banda de energía en los semiconductores. Un voltaje externo aplicado en la unión PN para su polarización directa, excita los portadores mayoritarios electrones, moviéndolos desde la banda de conducción del lado N a la banda de valencia del lado P. En el traslado los electrones cruzan la brecha de energía Eg, cediendo este potencial en forma de calor y de luz o fotones. Cada material semiconductor posee diferentes características de Eg, y la longitud de onda (λ), o sea el color de la luz emitida por el LED, depende de la magnitud de Eg, Pro-Lighting (2015).

Entre los últimos desarrollos que empiezan a aparecer en el mercado, están los LED de luz ultravioleta y el montaje de los LED en bases estándar de bombillos miniatura Edison E-14 y E27, para remplazar directamente los tradicionales bombillos 115 o 230 V.

Cuando la alimentación se derive de una fuente de voltaje constante, cada LED debe ser protegido conectándolo en serie con una resistencia limitadora de corriente. Los diodos emisores de luz deben ser alimentados exclusivamente por voltaje directo DC, por lo tanto un circuito con LED debería prever una protección contra el voltaje inverso si se anticipa que esto pueda exceder el voltaje inverso máximo del componente empleado, Pro-Lighting (2015).

El LED, necesita de poca corriente para trabajar, aproximadamente unos 20 miliamperios, puede decirse que toda la energía consumida se convierte en luz, alcanzando eficiencias cercanas al 100%. La ecuación siguiente nos permite calcular el valor de la resistencia limitadora de corriente que se ha de poner, para proteger un LED.

$$R_L = \frac{V_{CC} - V_F}{I_F}$$

$$R_L = \frac{12 - 1{,}5}{0{,}02} = \frac{18}{0{,}02} = 900\Omega$$

Como no es fácil conseguir resistencias de 900 ohmios se aproxima con una resistencia de 1000 ohmios, figura 18.

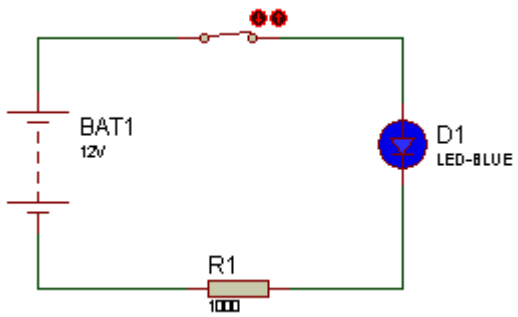


Figura 18. Esquema circuito diodo LED.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

2.5.3. Pantalla LCD 2x16

El funcionamiento de estas pantallas, se fundamenta en sustancias que comparten las propiedades de los sólidos y líquidos a la vez. Cuando un rayo de luz atraviesa una partícula de estos materiales tiene necesariamente que atravesar un cristal sólido, pero cada una de estas partículas se le puede aplicar una corriente eléctrica para que cambie su polarización dejando pasar la luz, Palacios (2008).

Una pantalla LCD figura 19, está formada por 2 filtros polarizados, colocados perpendicularmente, de manera que al aplicar una corriente eléctrica al segundo de ellos dejara pasar o no, la luz que ha atravesado el primero.

Figura 19. Fotografía de la pantalla LCD 2X16.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Normalmente la visión del carácter representado en el display, se produce por refracción de la luz en el mismo y tiene que ver con el ángulo desde donde se mire. La pantalla LCD, cuenta con 16 pines de conexión, de dos líneas de escritura donde se pueden mostrar 16 caracteres, Palacios (2008).

El Pin número 1 y 2, están destinados para la fuente de 5 voltios que requiere el módulo para su funcionamiento y el Pin número 3, es utilizado para ajustar el contraste de la pantalla; es decir colocar los caracteres más oscuros o más claros para poder observar mejor. El Pin numero 3: polarización del cristal. El Pin numero 4: denominado "RS" trabaja paralelamente al Bus de datos del módulo LCD.

Bus de datos, son los Pines del 7 al 14, este bus es utilizado de dos maneras, ya que se puede colocar un dato que representa una instrucción o podrá colocar un dato que tan solo representa un símbolo o un carácter alfa numérico. Pero para que el módulo LCD pueda entender la diferencia entre un dato o una instrucción se utiliza el Pin Numero 4. Si el Pin numero 4 = 0 le dirá al módulo LCD que está presente en el bus de datos una instrucción, por el contrario, si el Pin numero 4 = 1 le dirá al módulo LCD que está presente un símbolo o un carácter alfa numérico.

El Pin numero 5: denominado "R/W" trabaja paralelamente al Bus de datos del módulo LCD. Es utilizado de dos maneras, ya que usted podrá decirle al módulo LCD que escriba en pantalla el dato que está presente en el Bus; por otro lado podrá leer que dato está presente en el Bus. Si el Pin numero 5 = 0 el modulo LCD escribe en pantalla el dato que está presente el Bus; pero si el Pin numero 5 = 1 significa que usted necesita leer el dato que está presente en el bus del módulo LCD, Palacios (2008).

El Pin numero 6: denominado "E" que significa habilitación del módulo LCD, tiene una finalidad básica: conectar y desconectar el modulo. Esta desconexión Microprocesada no estará referida al voltaje que le suministra la corriente al módulo; la desconexión significa tan solo que se hará caso omiso a todo lo que esté presente en el bus de datos de dicho modulo LCD.

Los Pines desde el numero 7 hasta el número 14, representan 8 líneas que se utilizan para colocar el dato que representa una instrucción para el modulo LCD o un carácter alfa numérico. El Bus de datos es de 8 Bits de longitud y el Bit menos significativo está representado en el Pin número 7, el Pin más significativo está representado en el Pin número 14. Los Pines 15 y 16: estarán destinados para suministrar la corriente al Back Light, Palacios (2008).

3. DISEÑO Y DESARROLLO DEL PROYECTO

El prototipo es ideado para un piso plano, pues es bastante común encontrar bodegas destinadas a manufactura con dicha característica; en las instalaciones suponemos una temperatura ambiente de 25 grados Celsius.

Gracias a estos datos no se requiere de equipo especial para cumplir la función de desplazamiento, y los componentes electrónicos no deben tener sistemas de refrigeración forzada; entonces se considera el uso de materiales y repuestos de fácil consecución. En la construcción de la estructura se usa acrílico, que es fácil de trabajar, fuerte y liviano. El uso de ruedas es óptimo para el desplazamiento, es simple y confiable, se busca que las ruedas sean de buen agarre, un material recomendado es el neopreno.

Se planea que el prototipo transporte tornillos de 100 milímetros tamaño máximo y una cantidad máxima de 100 unidades, lo que da un peso aproximado de 3 kilogramos nominales la sola carga. Entonces los materiales del robot deben ser fuertes y livianos. En cuanto a la fuerza que los motores deben desarrollar, esta debe vencer la carga ejercida por el peso de los tornillos más la estructura del mismo prototipo, gracias a esto se deben usar motores con reductor de velocidad y aumento de torque.

Se adopta la opción de las baterías recargables, por ser más económicas en costo y son piezas intercambiables, además brindan autonomía de fuente al prototipo robot. Como cada día salen mejores versiones de baterías, se busca poder reemplazarlas de forma rápida, se implementa estuche porta pilas, y se adapta una entrada de voltaje para la carga de las baterías desde el exterior con un cargador de voltaje.

Debido a que el piso es plano, se piensa en la demarcación de una línea que sirva de guía para el prototipo, entonces se deben usar sensores reflectivos que diferencien contrastes de color, ya que es importante no alterar el aspecto físico de la fábrica. El prototipo debe tener una señal acústica que lo identifique en caso de una alarma en el tiempo de traslado.

3.1. ESTRUCTURA MECÁNICA

Al repasar las estructuras básicas, la que más se ajusta a la aplicación es la plataforma diferencial; esta reúne las características necesarias para el prototipo robot, gracias a que su cinemática le permite movimientos rotativos, avance en línea recta y giros a 90 grados.

El prototipo robot, tiene dos motores de corriente continua y cada uno por separado se acopla mecánicamente a un reductor de velocidad, que aumenta el torque para las ruedas de tracción. Cada rueda trabaja de manera independiente y tienen la capacidad de girar en velocidades diferentes. Un caster_ball en la parte delantera del robot, sirve de soporte y de dirección. Entre sus ruedas se ubica el eje cinemático, es aquí donde se realiza la diferencia en posición. La distribución de las ruedas, el caster_ball y los sensores se pueden apreciar en la figura 20.

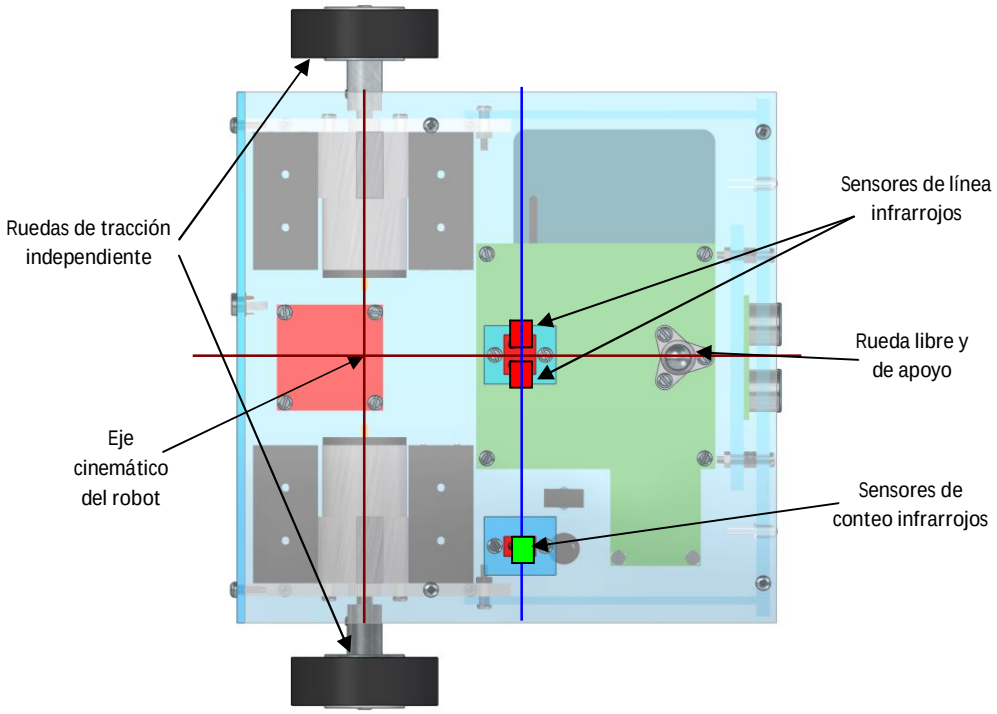


Figura 20. Dibujo de la plataforma vista por debajo.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Para el seguimiento de la línea, se dispone de dos sensores infrarrojos por debajo; otro sensor infrarrojo se usa para detectar las marcas de ubicación y un sensor por ultrasonido se usa para ubicar obstáculos en la trayectoria.

La figura 21, muestra una aproximación de la plataforma vista de lado, donde se aprecia la representación de la distribución de los componentes más relevantes del prototipo robot, incluyendo el sensor de ultrasonido y las baterías AAA.

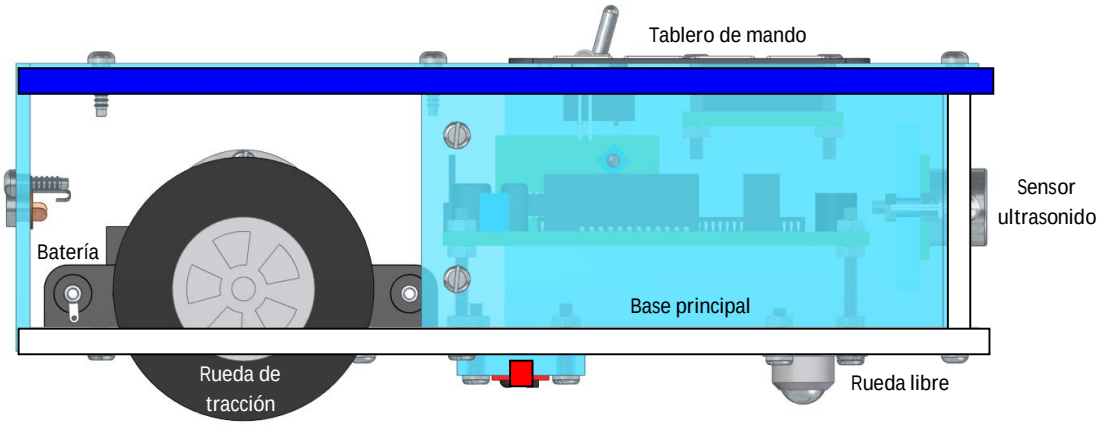


Figura 21. Dibujo de la plataforma vista de lado derecho.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

3.1.1. Elaboración de la plataforma

La fabricación de piezas mecánicas, está hoy en día destinada a máquinas CNC o máquinas de impresión 3D; pero tener a disposición estos equipos es realmente costoso y por ello se hace uso de recursos a la mano de nuestros pequeños talleres, donde se debe tener herramientas como: alicates, pinzas, destornilladores, bisturí, segueta y taladro.

Materiales:

- Láminas de acrílico, espesor de 5 milímetros, color azul traslucido y transparente.
- Tornillos de rosca fina, con paso 1 milímetro, de 3 milímetros de diámetro por 12 milímetros de largo.
- Tornillos golosos de 3 milímetros de diámetro y 12 milímetros de largo.
- Pegante instantáneo.

Herramientas:

Los alicates, los elegiremos de tal forma que sean lo más útiles posibles, si estos son de punta redonda sirven para doblar extremos de hilos de conexión. Los alicates de puntas planas ya sean de superficie interna lisa o grabada, sirven para agarrar o sujetar piezas que se unen con tornillos.

Las pinzas de muelle, son muy útiles para sostener los extremos de los hilos de conexión, son muy útiles para alcanzar objetos de difícil acceso, las hay con puntas recubiertas de capa de plástico o goma.

El bisturí, es un instrumento en forma de cuchillo pequeño de hoja fina, muy usado en artesanía, manualidades y en general en aquellas actividades o artes en que se requieren cortes finos y precisos.

Una segueta, es una herramienta cuya función es cortar o serrar, principalmente madera, aunque se usa para cortar láminas de metal y en este caso láminas de acrílico.

El taladro, es una herramienta con la que se realizan agujeros, gracias a la rotación de una broca movida por un motor eléctrico o neumático.

Limas redonda y plana, es una herramienta utilizada para el desgaste de piezas de distintos materiales, como: plástico, madera, metal y en este caso acrílico.

Regla o escuadra, herramientas que se usan para dibujo, vienen con escala grafica que les permite ser instrumentos de medición.

Papel lija, es una herramienta que consiste en un soporte de papel sobre el cual se adhiere como polvo esmeril, se usa para quitar fragmentos de material en superficies.

Consideraciones importantes

Es conveniente usar gafas de seguridad, para evitar que material particulado ingrese de manera accidental en los ojos. Cuando se realicen cortes de piezas es mejor usar guantes de baqueta, para evitar cortes en los dedos. Al momento de realizar los cortes se generan pequeñas partículas de acrílico, estas pueden ser aspiradas, es bueno usar mascarilla contra partículas o polvo.

Importante recordar que toda labor manual en ocasiones es peligrosa, si no se usan herramientas en buen estado.

Antes de realizar los cortes, se marcan las líneas usando lápiz de tinta indeleble, esto se hace con una regla o una escuadra, las láminas de acrílico se cortan con segueta, sujetándolas contra una superficie plana. Es conveniente usar una bayetilla o trapo de algodón debajo de la lámina y así evitar rayar la misma.

Para que las láminas de acrílico con dimensiones iguales queden simétricas, se les hacen orificios con el taladro y broca, se unen por medio de tornillos pasantes con tuerca, después se pulen con una lima plana. Con lija 320 se pulen las salientes filosas, para evitar cortes o lesiones involuntarias en los dedos.

De ser necesario se usan limas redondas para agrandar los agujeros y pulirlos. Las piezas que se deseen fijas, se pueden pegar con pegante instantáneo y las piezas que se deben soltar, usan tornillos de rosca ordinaria o comúnmente llamados golosos.

Unir piezas con tornillos es relativamente fácil, en el caso de los pasantes con tuerca, solo es necesario realizar los orificios y hacerlos coincidir, para que por ellos pase el tornillo y luego se pone la tuerca. Pero los golosos se incrustan en una de las piezas a unir.

Para instalar un tornillo goloso, se debe perforar un agujero con un diámetro inferior al del tornillo, el tornillo se calienta con el cautín y se atornilla en el orificio realizado. Las estrías de la rosca forman la superficie que retiene el tornillo, que al enfriarse dan soporte y no permiten que el tornillo se salga.

Construcción piezas A y B

La estructura mecánica del prototipo robot, se ha pensado en dos secciones de piezas A y B, que facilitan la explicación. La pieza **A1**, se construye a partir de una lámina de acrílico de 5 milímetros de espesor traslucida, con medidas de 200 milímetros de ancho y 200 milímetros de largo para la base principal; esta será donde se montarán los componentes electrónicos, la figura 22 muestra el acrílico recortado a las medidas antes enunciadas.

Figura 22. Fotografía del acrílico pieza A1.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La construcción de los soportes **A2**, para los motoreductores se desarrollan a partir de una lámina de acrílico de 5 milímetros de espesor, con medidas de 50 milímetros de ancho y 100 milímetros de largo, se unen con tornillos pasantes de rosca fina y tuercas, de manera que se puedan pulir y darle simetría a las piezas, como lo muestra la fotografía figura 23.

Figura 23. Fotografía de los soportes pieza A2.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Los motoreductores, son los encargados de mover las ruedas del robot, estos se fijan a los soportes rectangulares **A2**, además son acoplados a las ruedas de

neopreno por medio de un tornillo prisionero central y un tornillo rosca fina pasante con tuerca.

En la fotografía, figura 24 se muestra los soportes perforados y listos para fijarlos a la base, al igual la rueda de neopreno, esta última se instala sujetándola al eje del motoreductor con un tornillo prisionero.

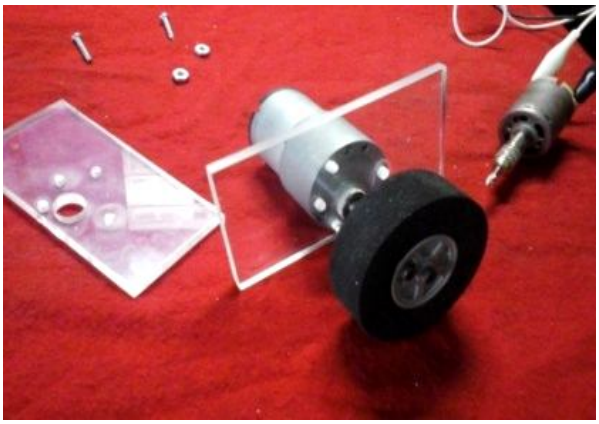


Figura 24. Fotografía de soportes terminados.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Se fija el soporte **A2**, a la base con pegante instantáneo y se instala el motoreductor y la rueda como lo muestra la fotografía figura 25.

Figura 25. Fotografía motoreductor en la base.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Luego de haber fijado los motoreductores, se tiene la base de la plataforma terminada, corresponde a la estructura **A**, de color transparente.

En la fotografía se ve la tarjeta electrónica del puente H, en medio de los motoreductores y el soporte esférico o caster_ball, en la parte delantera de la plataforma, además se puede apreciar de forma clara la cinemática del prototipo robot, figura 26.

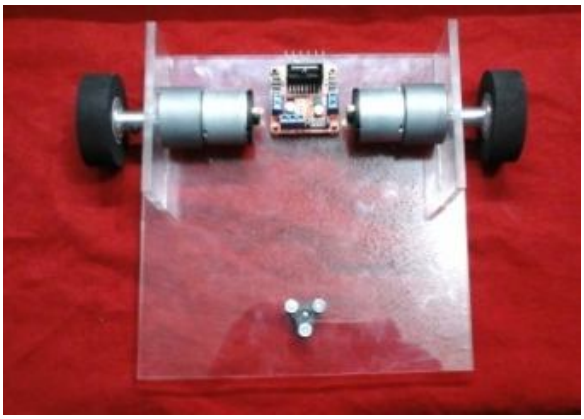


Figura 26. Fotografía de la base A terminada.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

El cuerpo del prototipo robot, es donde se instalan los componentes con los que interactúa el operador; el teclado matricial, usado para ingresar datos al controlador y la pantalla LCD, usada para visualizar acontecimientos.

En la parte delantera del prototipo robot se instala el sensor de ultrasonido, que servirá para detectar obstáculos en la trayectoria. La pieza que tendrá esta tarea se nombra **B2** y tiene una altura de 50 milímetros, largo de 195 milímetros y espesor de 5 milímetros. En ella se perforan dos orificios de diámetro de 15 milímetros, por donde salen los transductores del sensor de ultrasonido, como lo muestra la figura 27.

Figura 27. Fotografía de la pieza frontal B2.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La pieza **B1**, es la plataforma de carga y el panel frontal en este prototipo; teniendo en cuenta que el prototipo robot se fabrica con fines de investigación se le resta importancia como herramienta de trabajo industrial y se profundiza en los aspectos de diseño hardware y software.

Las dimensiones de la pieza **B1** son iguales a las de la pieza **A1**; lámina de acrílico de 5 milímetros de espesor traslucida azul, con medidas de 200 milímetros de ancho y 200 milímetros de largo. En ella se perfora un rectángulo de 25 milímetros de ancho y 70 milímetros de largo, para que por aquí salga la pantalla LCD, además se perforan un orificio de 5 milímetros de diámetro para el interruptor de codillo y un rectángulo de 3 milímetros de ancho por 20 milímetros de largo para la cinta del teclado matricial y los orificios para los tornillos, la figura 28 muestra el teclado matricial sobre la pieza **B1**.

El teclado matricial, se instala sobre el acrílico usando el adhesivo que tiene por el revés, donde se debe tener especial cuidado de no ensuciar la superficie, o el teclado no sentará bien y con el tiempo se cae.

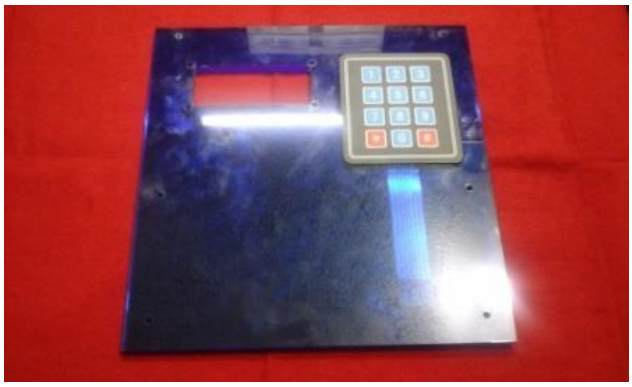


Figura 28. Fotografía de la pieza B1.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

De manera similar como se instaló los motoreductores, se sujetan los componentes electrónicos de prototipo robot, por ejemplo la porta pilas, la pantalla LCD, el circuito puente H, se unen a la plataforma con tornillos pasantes y tuercas. La instalación de la porta pilas, se hace conveniente en este momento, ya que de aquí en adelante se determinará el espacio disponible para los circuitos electrónicos, la posición de las mismas se debe a que si el peso se acerca más al eje cinemático del prototipo robot, este no presentará inestabilidad en sus movimientos.

La figura 29, muestra la instalación de la pieza **B2** con el sensor de ultrasonido y dos LEDs bicolor en el panel delantero. En la foto se aprecia la instalación de los porta pilas al lado de los motoreductores. Además se ve una lámina de acrílico que sujeta el sensor de ultrasonido por detrás, esta tiene 30 milímetros de ancho y 100 milímetros de largo, fijada con tornillos de rosca fina de longitud 20 milímetros pasantes con tuerca.

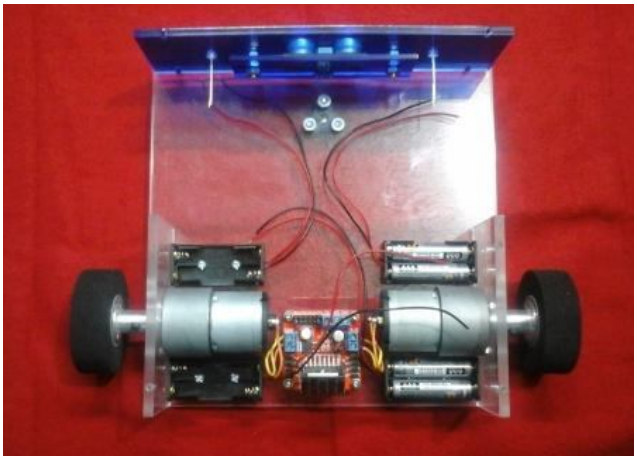


Figura 29. Fotografía de instalación pieza B1.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La figura 30, muestra las piezas **B1** y **B2** con el teclado matricial instalado, la pantalla LCD fijada con tornillos pasantes y tuercas, el interruptor de codillo sujetado a través del acrílico con tuerca y el sensor de ultrasonido, todo sobre la plataforma **A1**.

Figura 30. Fotografía de la plataforma.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La figura 31, muestra los dibujos de las piezas en acrílico, que forman la plataforma del prototipo robot. Las piezas A2 y B4, son por dos unidades.

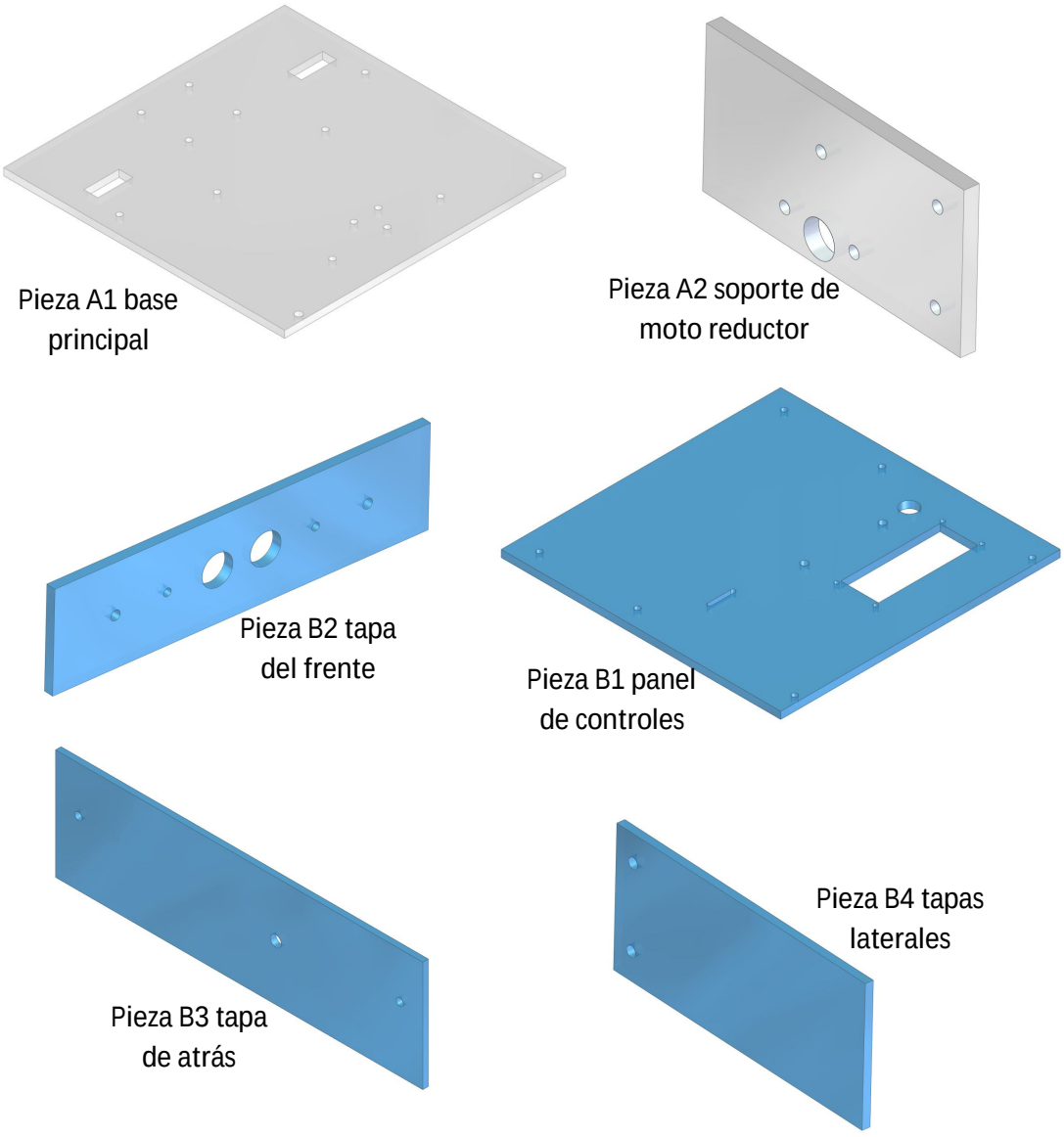


Figura 31. Dibujos de las piezas que forman a estructura.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Los dibujos fueron realizados con el programa Siemens Solid Edge ST7, que permite descargar una licencia educativa gratis por un año.

https://www.plm.automation.siemens.com/es_sa/academic/resources/solid-edge/student-download.cfm

3.2. CIRCUITO ELECTRÓNICO

El circuito electrónico del prototipo robot, consta principalmente de la tarjeta PCB controladora. Donde se ha considerado de vital importancia el poder cargar el programa en el microcontrolador, sin retirar el circuito integrado de la tarjeta PCB, ya que evita daño del mismo y agiliza las pruebas de los programas. Por tanto se incluye la conexión al grabador como parte básica del circuito. Se usa conector de espadines para insertar una extensión al dispositivo grabador.

Como se explicó anteriormente, para que funcione el microcontrolador ejecutando una tarea se necesita un programa; el circuito integrado por si solo no sirve para nada, requiere de la información necesaria con instrucciones lógicas. Esta información se carga en el integrado por medio de un aparato llamado grabador o quemador y para este proyecto se usa el **Pickit 2** de Microchip. La figura 32, muestra los seis pines del grabador, por donde se carga el programa a la memoria FLASH del microcontrolador.

Para conectar el grabador o quemador al microcontrolador, se debe seguir las indicaciones de la hoja de datos proporcionada por el fabricante, ya que este ha asignado pines para la grabación de los datos en el microcontrolador.

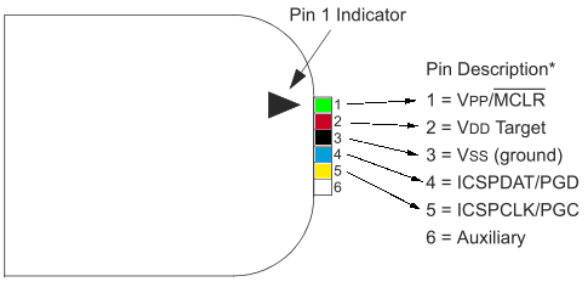


Figura 32. Pines de salida programador Pickit 2.

Fuente: Hoja de datos Pickit 2 Microchip Technology Inc ©.

Como recomendación especial, se debe tener en cuenta no energizar el microcontrolador si está conectado el grabador, pues este voltaje puede dañar los circuitos internos de este último, para evitar daño siempre desconecte el programador de la tarjeta PCB si se desea hacer pruebas. Antes de cargar algún programa, retire el voltaje de alimentación del circuito, se recomienda instalar un LED que indique presencia de voltaje.

Cada uno de los pines del grabador está destinado para una función específica:

- **Pin 1** del programador conectado al **Pin 4** RA5/MCLR del microcontrolador.
- **Pin 2** del programador conectado a 5 voltios o positivo.
- **Pin 3** del programador conectado a 0 voltios o negativo.
- **Pin 4** del programador conectado al **Pin 13** RB7/PGD del microcontrolador.
- **Pin 5** del programador conectado al **Pin 12** RB6/PGC del microcontrolador.
- **Pin 6** del programador no usado.

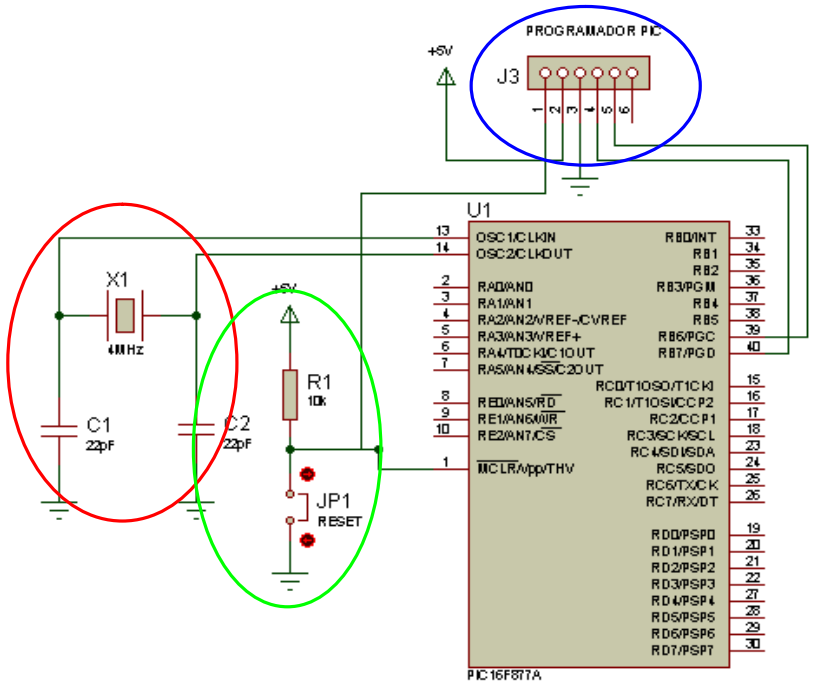


Figura 33 Esquema microcontrolador conexionado a grabador.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

En la figura 33 se muestra el esquema básico, sugerido por el fabricante del microcontrolador: el óvalo azul, corresponde a el esquema de conexión entre el microcontrolador y el grabador; el óvalo rojo, corresponde a la conexión del oscilador y el óvalo verde, corresponde a el esquema de reinicio. Este es el circuito inicial dentro del proyecto.

Se puede apreciar, que los puertos quedan disponibles para las aplicaciones que se requieran, en el caso del puerto B nada afecta tener los pines RB6 y RB7 conectados a la terminal del grabador, ya que solo se conecta este último cuando se necesita cargar los programas en el microcontrolador y el resto de tiempo no está presente esta conexión.

Partiendo del esquema básico, se deben asignar al microcontrolador en cada uno de sus puertos los elementos de entrada como: el teclado, los sensores reflectivos, el sensor de ultrasonido y los elementos de salida como: la pantalla LCD, el buzzer, el circuito de control de los motores puente H, los LEDs. Para que por medio de la programación este realice las funciones que le han sido asignadas.

Con el teclado matricial de 4X3, se le introducen datos de tipo carácter, usados para direccionar el comportamiento futuro del prototipo robot, por ejemplo decir a donde debe ir. Estos datos interactúan con la programación cargada en el microcontrolador.

Los sensores, se encargan de informar al microcontrolador sobre acontecimientos externos, ya sean los tres sensores de tipo reflectivos infrarrojos o el sensor de ultrasonido. Los sensores infrarrojos le dirán la posición del prototipo frente a la línea a seguir, así como la ubicación referente a las estaciones, y el sensor de ultrasonido le dirá si se presenta algún objeto frente al prototipo robot.

La pantalla LCD, permite interactuar con el prototipo robot, en ella se dan indicaciones de manejo por medio de mensajes, por ejemplo: indica distancia de un objeto al prototipo. El buzzer, sirve para indicar de manera audible acontecimientos como por ejemplo: fallas al ingresar datos y presencia de objetos, entre otras acciones, además ayuda al operador en el proceso de manipulación del teclado.

El microcontrolador, luego de procesar la información recibida de los sensores, se encarga de gobernar al circuito de polarización de los motores, que es conocido como puente H, este último actúa directamente sobre los motores para que giren en un sentido u otro, dando movimiento al prototipo robot.

Al microcontrolador se le conectan los componentes enunciados y de esta forma junto con la programación, se crea el cerebro del prototipo robot.

La figura 34, muestra el esquema del microcontrolador conectado al teclado matricial y a la pantalla de LCD; se puede ver que tanto la pantalla como el teclado están conectados a un mismo puerto. Esto se realiza a causa de tener mayor disponibilidad de pines para las demás conexiones a otros componentes.

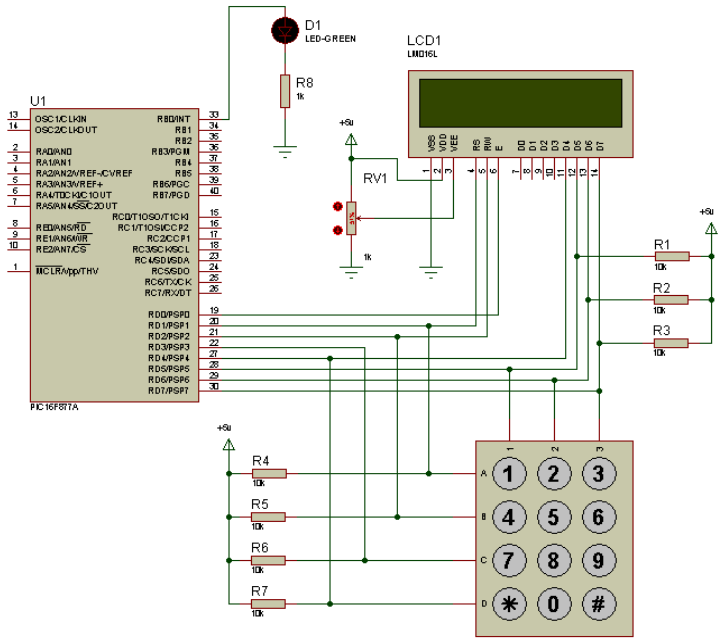


Figura 34. Esquema del circuito LCD y teclado.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

3.2.1. Esquema electrónico en Proteus ISIS

Una forma de diseñar circuitos electrónicos, para ensayarlos y cambiarlos rápidamente, antes de llevarlos al montaje físico es usando un simulador. De esta manera si se comenten errores de polarización, sobre carga o malos cálculos, los componentes no se destruyen y se pueden realizar cambios sin tener que gastar ningún recurso y además rápidamente.

El programa de diseño Proteus, ofrece la posibilidad de simular código en el microcontrolador, esto permite diseño tanto a nivel de hardware como de software en un mismo entorno. ISIS es el entorno grafico para la simulación y ARES es el entorno para el diseño de las tarjetas electrónicas PCB. Posee una muy buena colección de librerías de modelos, para dibujar, simular y crear placas, además permite la creación de nuevos componentes.

Para iniciar a trabajar con Proteus se debe abrir el programa, dando doble clip en el ícono del escritorio, como lo muestra la figura 35.

Figura 35. Icono de Proteus en escritorio.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Aparece el entorno de trabajo de Proteus, figura 36.

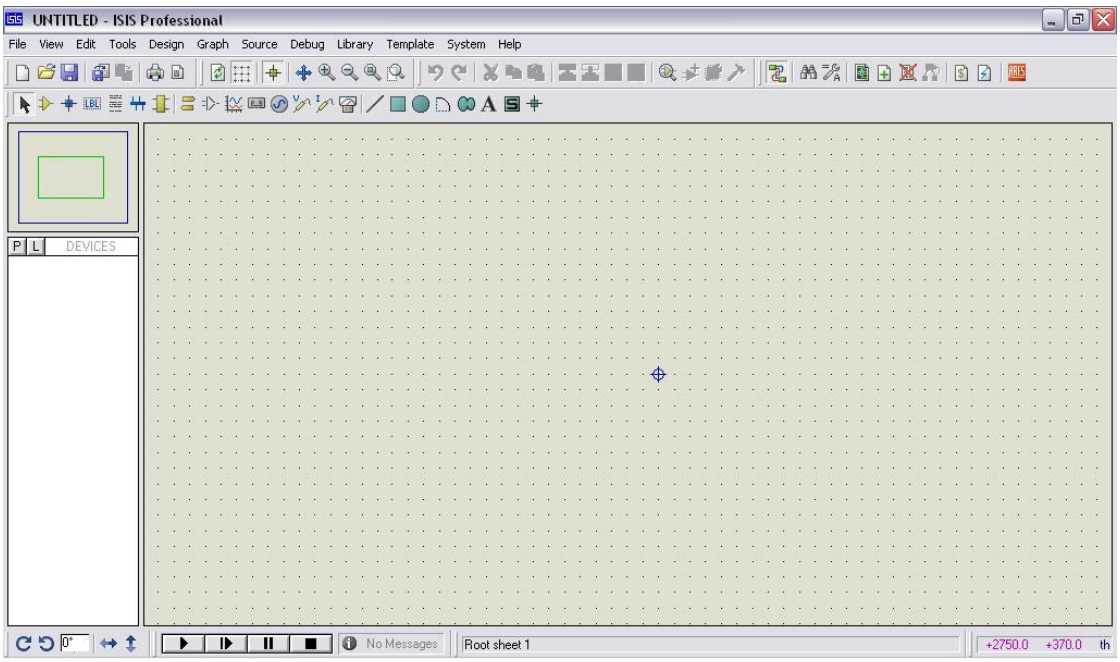


Figura 36. Ventana entorno de trabajo en Proteus.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Al pulsar el botón derecho del ratón, en cualquier parte del entorno de trabajo aparece un menú contextual de donde se pueden ir obteniendo los distintos submenús, figura 37.

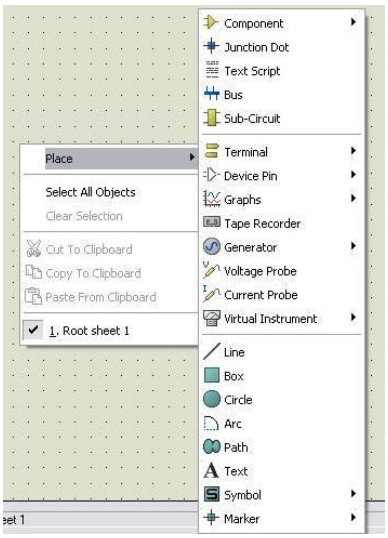


Figura 37. Submenús de trabajo Proteus.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Si se desea dibujar, lo primero es colocar los distintos componentes en la hoja de trabajo, esto se realiza seleccionando el modo componentes, figura 38 óvalo azul, luego dar clip sobre el botón P de la ventana de DEVICES figura 38, óvalo rojo.

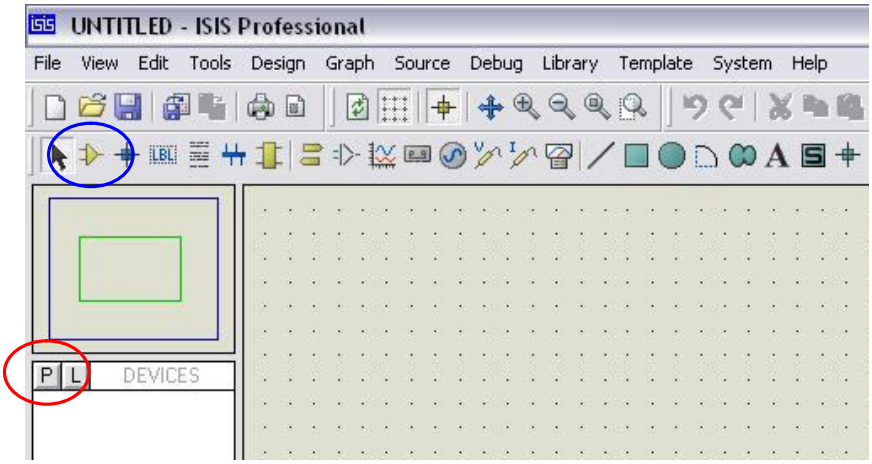


Figura 38. Selección componentes en Proteus.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Luego al activar el botón P se abre la ventana para la edición de componentes, figura 39, donde se puede buscar el adecuado y comprobar sus características.

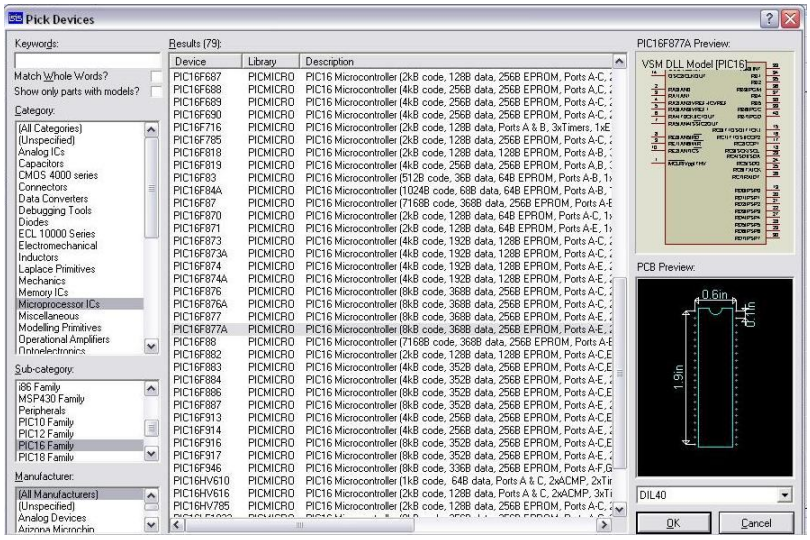


Figura 39. Ventana edición componentes.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Al localizar el componente que se necesita, se realiza una doble clip en él, de tal forma que aparezca en la ventana de DEVICES, se puede realizar esta acción tantas veces sea necesaria, así como componentes se quieran incorporar al esquema, figura 40. Una vez finalizado el proceso se puede cerrar la ventana de edición.

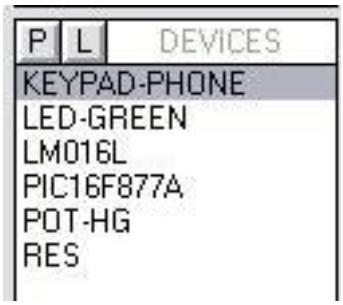


Figura 40. Lista de componentes añadidos.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Para situar un componente en el esquema, sólo debemos seleccionarlo de la lista, realizar un clip sobre la ventana de trabajo y se colocará automáticamente, y el cursor del ratón se convierte en un lápiz blanco. Se pueden colocar varios componentes del mismo tipo simplemente realizando varios clips. Para terminar de colocar un componente se debe seleccionar otro en la lista o pasar a otro modo de trabajo.

Una vez situados los componentes en el área de trabajo, se pueden mover de lugar, al pasar por encima del mismo, el cursor se convierte en una mano y al realizar un clip, el cursor se transforma en una mano con una cruz, indicando que se puede mover el componente, si se vuelve a realizar otro clip con el botón izquierdo se editan las características del componente.

Se puede cambiar su orientación, utilizando los comandos de rotación y reflexión a través de un clip del botón derecho del ratón figura 41 y para eliminar con dos pulsaciones con el botón derecho sobre el componente o con el botón derecho y el comando *Delete Objet.*

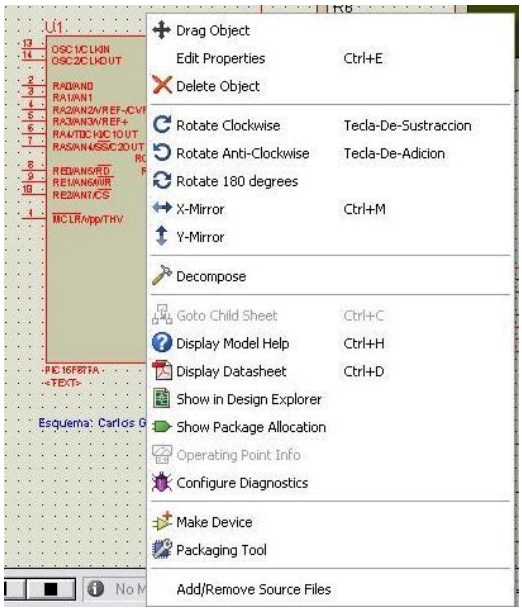


Figura 41. Menú del componente activado.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Todas estas acciones se pueden realizar individualmente o de forma colectiva, es decir, se pueden agrupar varios componentes, con dar dos clips consecutivos sobre ellos y manteniendo la tecla control pulsada o dibujando una ventana con el botón izquierdo y arrastrándola sobre los mismos.

Para unir los componentes con cables, hay que situarse en los extremos de los terminales, el cursor se convierte en un lápiz verde, se da clip y se va marcando el camino hasta el destino, o se puede realizar una pulsación en el inicio, luego en el destino, y dejar que ISIS realice el camino.

La figura 42, muestra el esquema realizado como se indico hasta aquí.

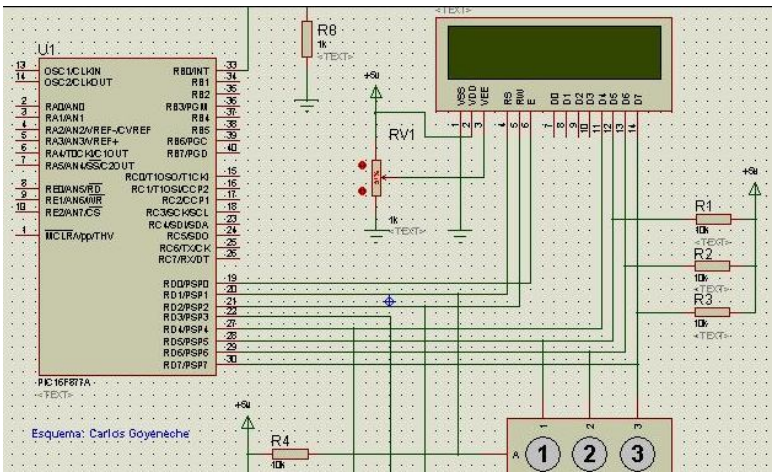


Figura 42. Esquema del circuito con componentes unidos.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Sólo queda modificar las características de los componentes que lo requieran, por ejemplo el valor de los componentes pasivos. Para ello, se selecciona un componente realizando doble pulsación con el botón izquierdo y aparece Edit Componet, como lo muestra la figura 43.

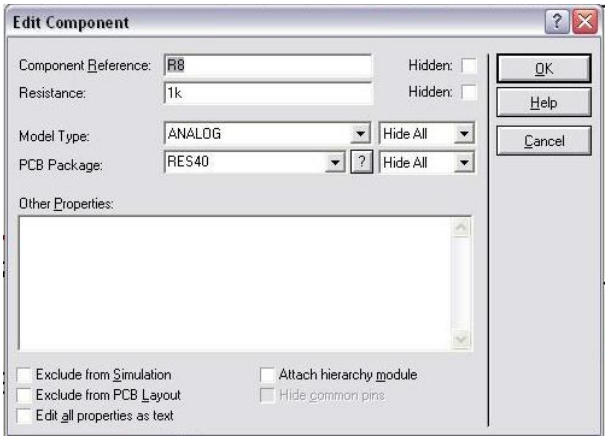


Figura 43. Ventana de edición de componente.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Edit Componet, es donde se pueden cambiar las características de los componentes, por ejemplo: la referencia o el valor de una resistencia en ohmios.

Por último se guarda el esquema realizado, para el caso de este proyecto, se le asigna el nombre de CONTROLADOR.DSN. La figura 44, muestra el circuito electrónico terminado y exportado a formato .bmp desde Proteus a partir de realizar la acción de exportar como mapa de bits, aquí se aprecia el microcontrolador conectado al teclado matricial, a la pantalla LCD, al circuito puente H y los sensores se representan por interruptores de dos posiciones.

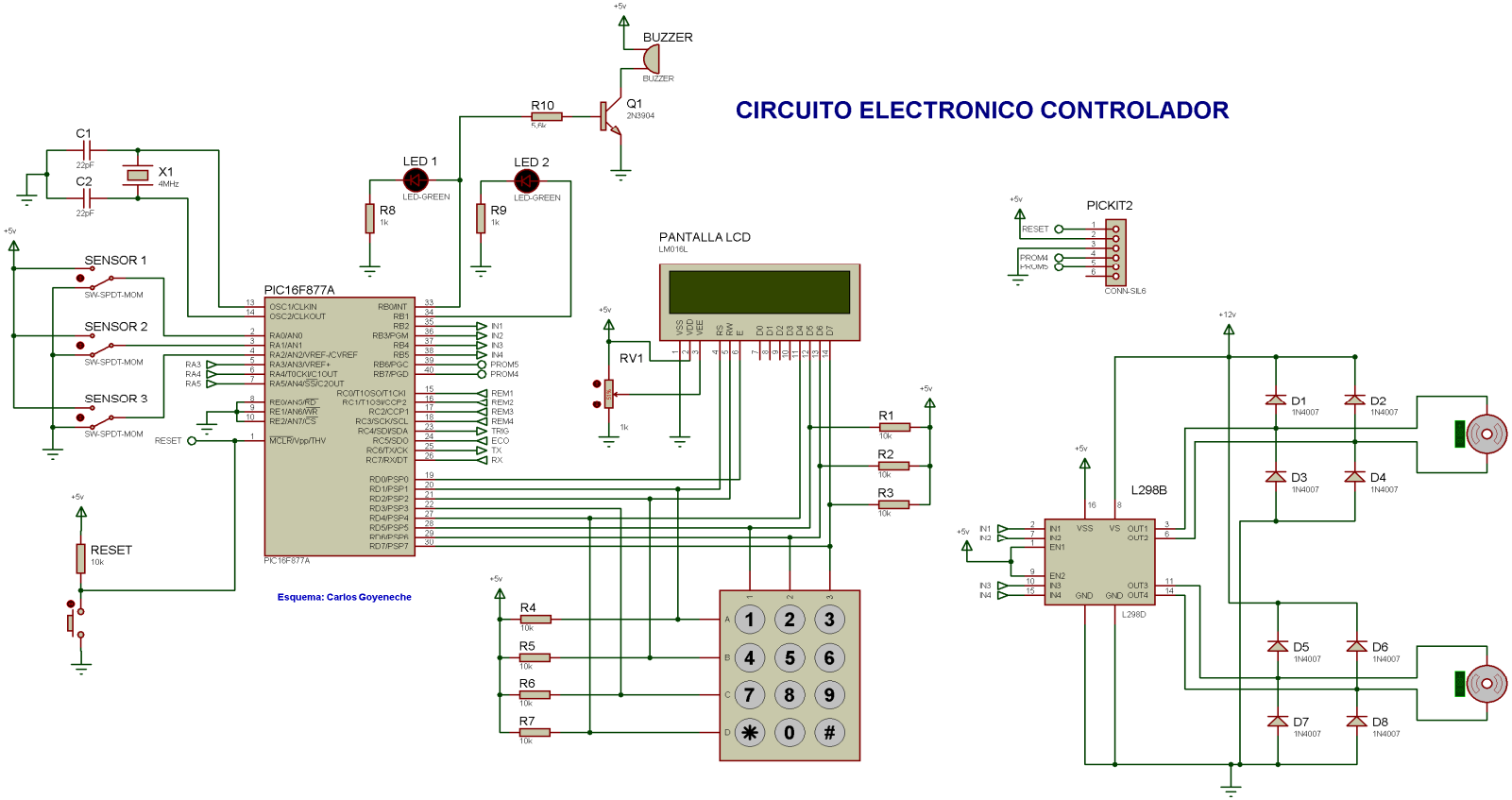


Figura 44. Esquema del circuito electrónico para simulación.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

3.2.2. Descripción del circuito electrónico

Se ha determinado trabajar con un microcontrolador PIC16F877A gracias a sus grandes ventajas en la elaboración de proyectos; gran cantidad de información y tutoriales, facilidad para adquirir el circuito integrado, bajo costo, permite modificar su configuración cableada y programación interna sin realizar cambios representativos del circuito electrónico, ocupa poco espacio, consume poca energía, es rápido, preciso, tiene buena memoria para almacenar datos y capacidad de procesamiento, por ultimo requiere de muy pocos componentes externos para su puesta en servicio; esto simplifica el trabajo de diseño de tediosos circuitos electrónicos.

Figura 45. Diagrama de bloques sistema electrónico.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Una de las ventajas de tener un microcontrolador, es que se pueden usar los puertos para cualquier aplicación básica; gracias a ello si se requiere instalar un interruptor a cambio de un sensor y configurarlo como elemento de entrada, solo es necesario modificar la programación y asignarle la función específica.

Dos sensores infrarrojos de referencia QTR1A análogos, que se representan en la figura 45 con bloques de color rojo, están destinados a informar de la posición del prototipo con respecto a la línea a seguir, esta última se dispone en contraste de color negro. Los sensores son ubicados en los pines RA0 y RA1 del microcontrolador y configurados como entradas.

El seguimiento de línea es posible, a causa que se tienen en cada sensor un diodo emisor de luz infrarroja y un fototransistor que recibe la refracción de esta luz infrarroja y la traduce en voltaje eléctrico, este valor de voltaje es llevado al microcontrolador y por medio de programación son procesados de manera digital.

Tiene un sensor infrarrojo QTR1A digital, que se representa en la figura 45 de color verde; sirve para contabilizar las veces que se ha encontrado marcas, indicando la posición con respecto al punto de partida o la estación y es conectado al pin RA2 del microcontrolador y configurado como entrada.

El sensor de ultrasonido de referencia SRF04, representado en la figura 45 de color azul; está conectado al microcontrolador en el pin RC4 para el disparo configurado como salida y conectado al pin RC5 para el eco configurado como entrada, se encarga de verificar si frente al prototipo se encuentra algún obstáculo. En tal caso, el programa del microcontrolador ordena detener los motoreductores si detecta algún objeto y da una alarma a tal suceso por medio del buzzer; cuando no encuentra obstáculo deja avanzar en busca de su destino de manera normal.

El buzzer, se ha conectado al pin RB0 del microcontrolador y el pin se configura como salida, en la figura 45 el buzzer es representado de color verde oscuro.

La pantalla LCD, en la figura 45 de color azul oscuro y el teclado matricial en la figura 45 de color negro, son conectados al puerto RD en paralelo y todos sus pines son multiplexados por programa. Le permiten al usuario interactuar con el prototipo robot, ingresando datos y la pantalla le permite saber lo que ocurre en un momento dado.

Los motoreductores de corriente directa, están acoplados cada uno a una rueda de neopreno; en su activación se usa un circuito puente H, representado de color café en la figura 45, conectado a cuatro pines configurados como salida del microcontrolador RB2, RB3, RB4 y RB5. Es conocido que el sentido de giro de un motor de corriente continua, depende de la polaridad que se le aplica a sus terminales, en consecuencia para cambiar el giro es necesario intercambiar los terminales del motor o bien cambiar la polaridad de la alimentación. La mejor manera de controlar un motor de corriente continua de baja potencia, en velocidad y sentido de giro, es mediante la conmutación electrónica con el circuito puente H, el usado para este proyecto es el L298B.

El control remoto permite al operador cancelar una actividad, dar una alarma y tener control manual. Esto se logra por medio de la unidad de radio frecuencia RF M40-RFC representada en el diagrama de la figura 45 con color amarillo, conectado a los pines de microcontrolador RC0, RC1, RC2 y RC3, configurados como entradas.

Para que todo el sistema electrónico funcione se hace necesaria la fuente de alimentación, en este prototipo se usan baterías recargables AAA, representadas en la figura 45 con color naranja.

Se realiza el montaje del circuito en protoboard, para probar el funcionamiento de la programación, esto debido a que el simulador Proteus no tiene algunos componentes electrónicos y que los tiempos de simulación no corresponden a los de trabajo real. En la figura 46 se ve el circuito electrónico cableado y listo para experimentar. Se tiene el grabador **Pickit 2** óvalo color blanco, el teclado matricial óvalo color negro, el circuito puente H óvalo color café, la pantalla LCD óvalo color azul oscuro, el sensor de ultrasonido óvalo color azul, los sensores infrarrojos óvalo color rojo y el microcontrolador PIC óvalo color verde.

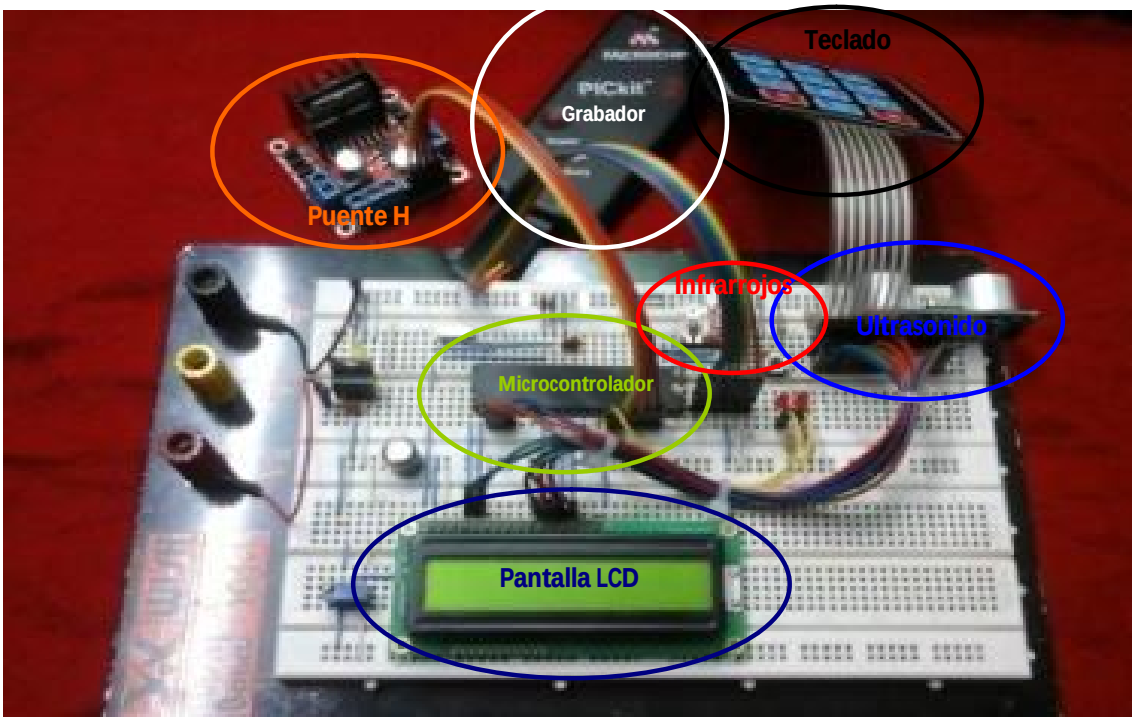


Figura 46. Fotografía del circuito cableado en protoboard.

Fuente: Goyeneche, C. (2016) Datos de estudio.

3.2.3. Diseño del PCB en ARES de Proteus

Una vez hecho el esquema del circuito en Proteus, para la simulación se procede a realizar una copia del archivo .DSN anteriormente creado, se le cambia de nombre para el caso en este proyecto se ha cambiado por CONTROLADOR_PCB.DSN y se realizan algunas modificaciones que corresponden a retirar los componentes de la lista de componentes añadidos que no tienen PCB y reemplazarlos por conectores con el mismo número de terminales, por ejemplo el teclado matricial no tiene PCB en el simulador Proteus y el usado en físico tiene una conexión de 7 pines hembra, por lo tanto se puede usar un conector CONN-SIL7 como el mostrado en la figura 47, que tiene PCB. Al realizar estos cambios se debe exportar el proyecto al programa ARES.

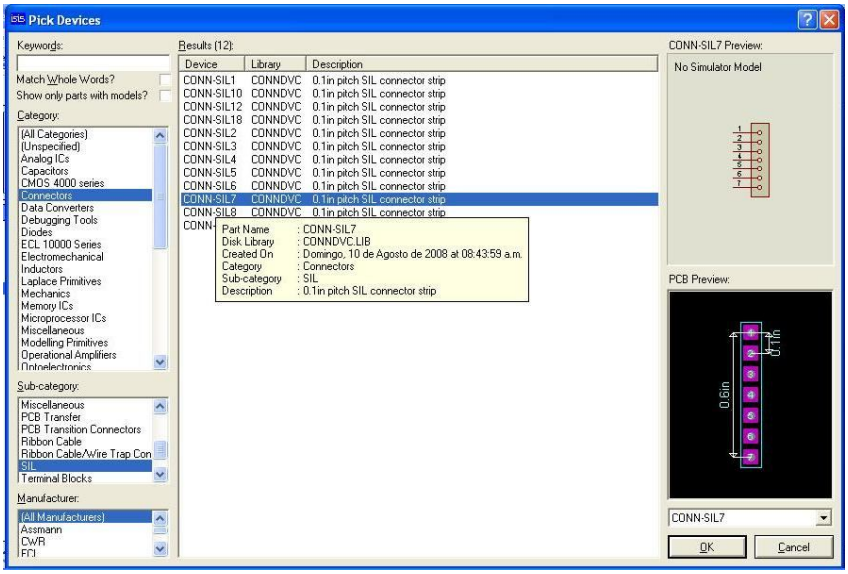


Figura 47. Ventana de edición componentes en ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Algo que se debe tener en cuenta, es referenciar bien las fuentes de voltaje, por ejemplo POWER para alimentación positiva en el caso 5 voltios positivos y GROUND para referencia cero o tierra, de no hacerlo así el programa no conectara los terminales de alimentación del microcontrolador y no se les crearan conexión en el PCB.

Importante recordar, que una vez exportado el circuito desde ISIS a ARES, los cambios que se realicen luego, no son tomados en cuenta en el programa ARES.

La figura 48, muestra como ha quedado el esquema electrónico en ISIS para exportarlo a ARES y generar el PCB. El esquema no es muy estético, pues muestra todas las conexiones entre los pines de los componentes, esto se debe realizar ya que el compilador del programa ARES solo creará las pistas basado en estas conexiones.

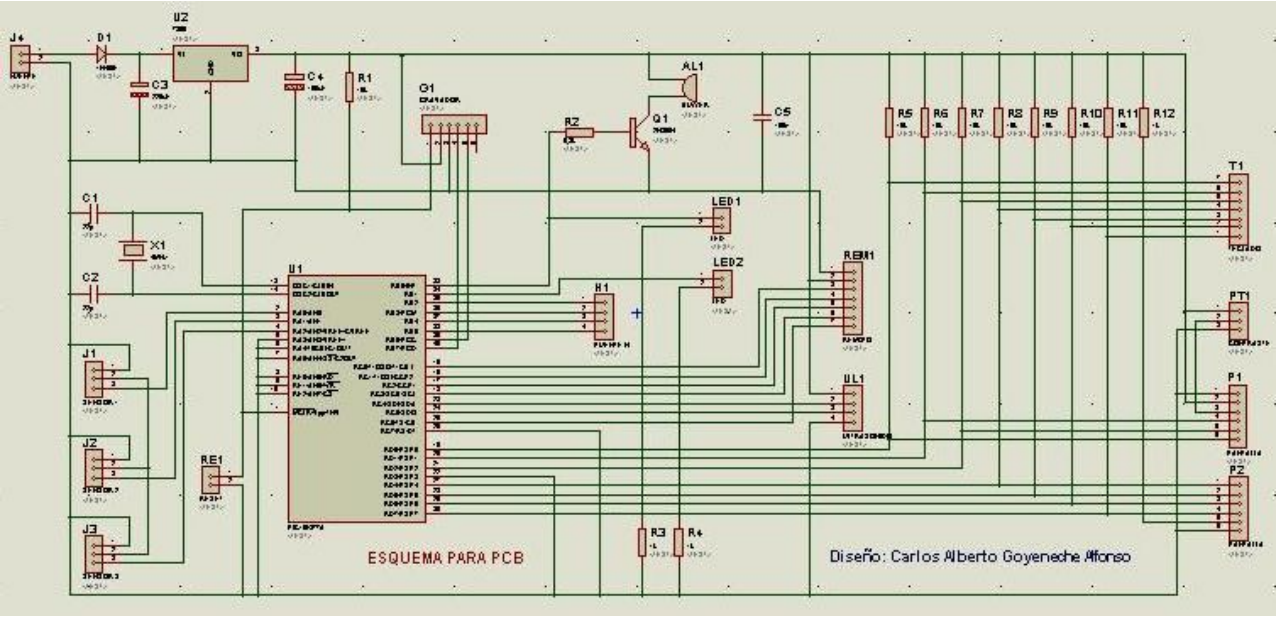


Figura 48. Esquema con cambios para exportar a ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Para realizar la transferencia desde el ISIS del proyecto, se da clip en el icono marcado como ARES de la barra de herramientas, mostrado en la figura 49, encerrado en el óvalo rojo, nombrado como **Netlist Transfer To ARES**.

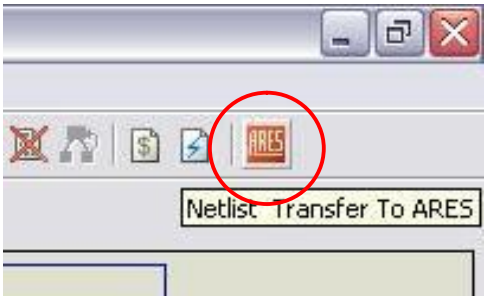


Figura 49. Icono para transferencia a ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Luego de realizar esta acción aparece la siguiente ventana, mostrada en la figura 50. Donde se ve el entorno de trabajo en ARES, los iconos de la barra de herramientas y las diferentes opciones de diseño, es aquí donde se realizan los PCBs.

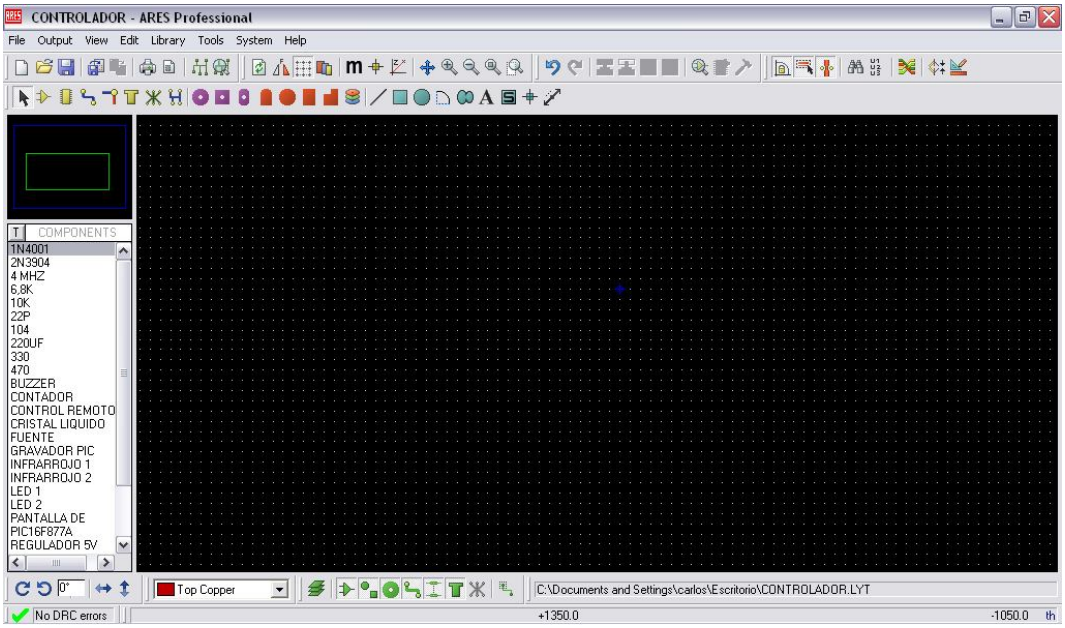


Figura 50. Ventana de trabajo en ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Una vez en ARES se debe asignar el área donde se han de montar los componentes de la tarjeta, esta acción se realiza al escoger en la barra desplegable ubicada en la parte de debajo de la ventana de trabajo, marcada como **Board Edge,** figura 51.

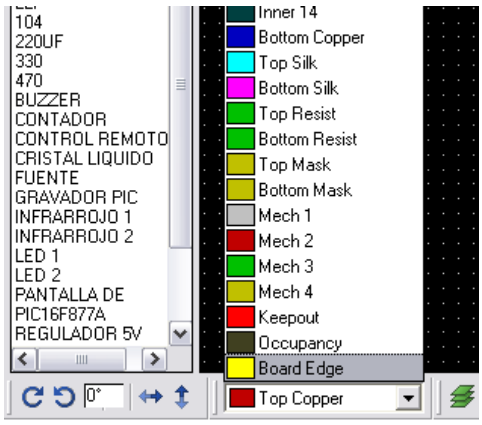


Figura 51. Ventana de trabajo en ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Primero se posiciona el puntero en la ventana de trabajo y al dar clip se marca el área donde se ubicarán los componentes, esto se puede ver en la figura 52.

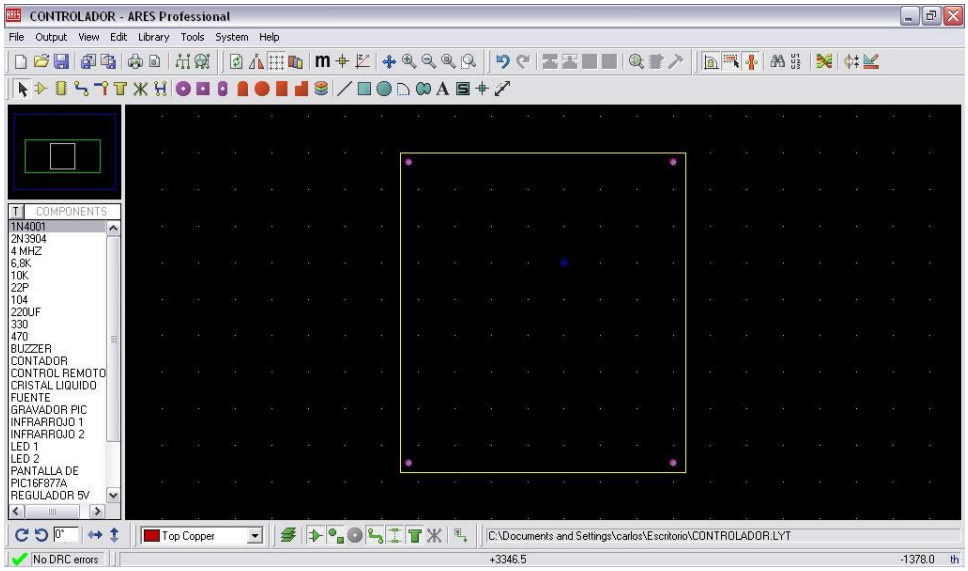


Figura 52. Área de la PCB en ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Para traer los componentes al área dibujada, seleccionamos en el menú **Tools** la opción **Auto Placer,** la figura 53, muestra cómo hacerlo.

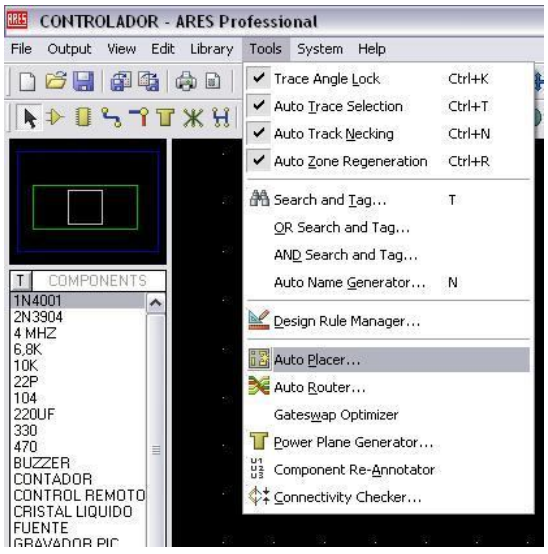


Figura 53. Menu Tolls opción Auto Placer.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Luego de dar clip aparece la siguiente ventana, figura 54, donde se ve la lista de los componentes resaltados con un visto bueno que indica que si se pueden añadir, en caso contrario estos no pueden pegarse al área de trabajo y se debe modificar en ISIS el diseño y buscar componentes de dimensiones similares. En esta ventana se puede escoger la dirección de montaje para los componentes (horizontal o vertical), por último se da OK.

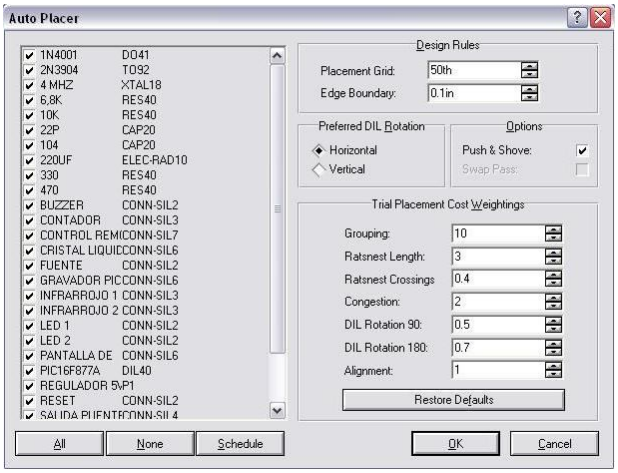


Figura 54 Ventana de opciones de pegado en ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Al realizar lo anterior, el software hace el pegado de los componentes de manera arbitraria, la figura 55, muestra cómo queda el diseño.

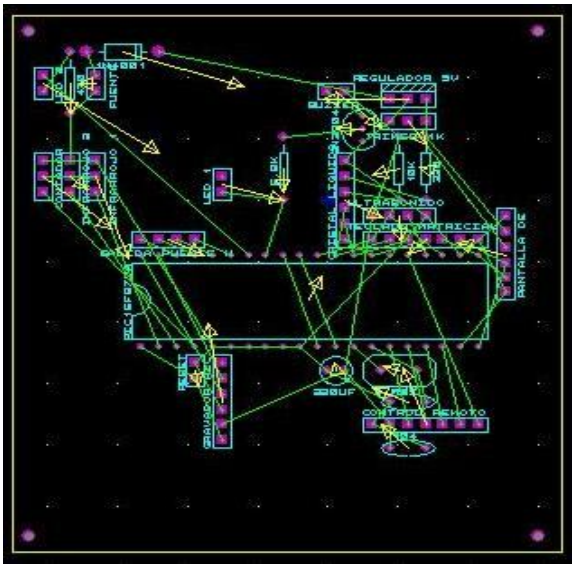


Figura 55. Ubicación con Auto Placer en ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

A partir de este paso, se inicia la organización de los componentes dando estética al PCB y buscando que las pistas sean fáciles de realizar. Los componentes se pueden mover dando clip derecho sobre ellos mismos, como lo muestra la figura 56.

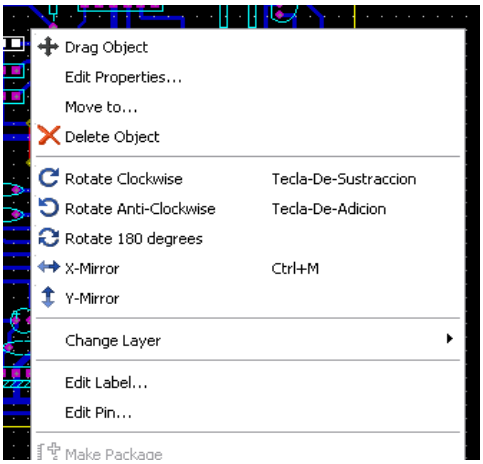


Figura 56. Ventana de funciones para los componentes en ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

La figura 57, muestra terminado el posicionamiento de los componentes en el área de la tarjeta, es aquí donde podemos visualizar cómo será el resultado final del diseño.

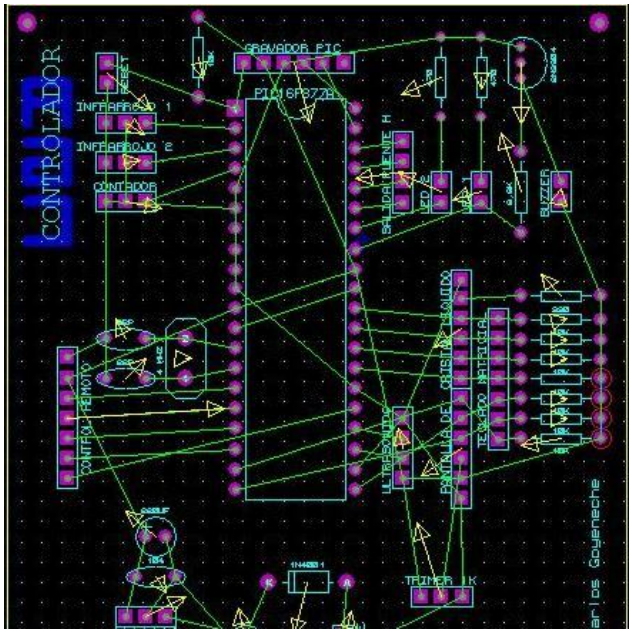


Figura 57. Posición de los componentes en ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Se realiza el **Auto Router**, que se encarga de dibujar las pistas siguiendo los trazos de las conexiones realizadas en el programa ISIS, la figura 58 muestra donde aparece la opción.

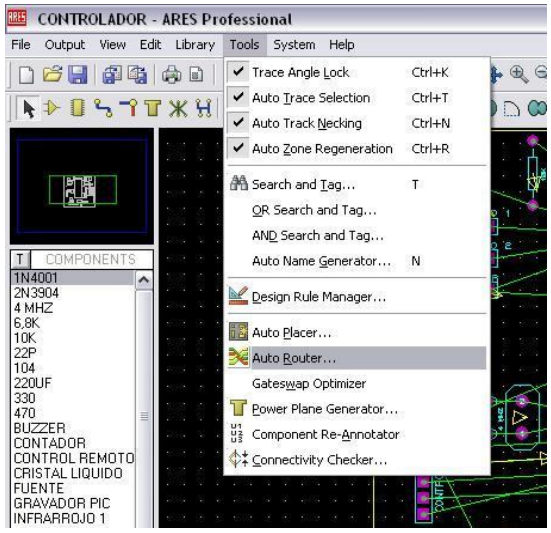


Figura 58. Menú Tools opción Auto Router.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Al dar clip en Auto Router aparece la siguiente ventana, figura 59, con las opciones de configuración para las pistas.

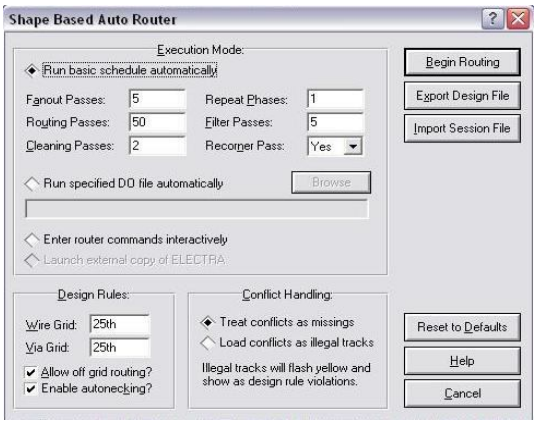


Figura 59. Ventana de propiedades de Auto Router.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Se da clip en **Begin Routing** y este dibuja las pistas. Si se desea modificar alguna pista se debe seleccionar la misma con el clip derecho del Mouse y se escoge la opción **Change Route**.

Al terminar de realizar los cambios y mejoras convenientes el diseño queda como lo muestra la figura 60, si se desea realizar modificaciones se recomienda realizar una copia y en ella trabajar, ya que de presentarse un error no se pierde el trabajo hecho.

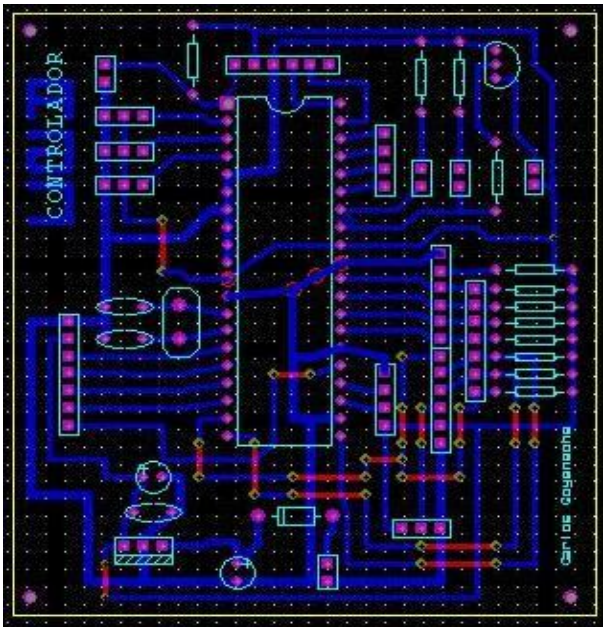


Figura 60. Vista del diseño de la PCB en ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

ARES cuenta con la opción de visualizar el diseño en 3D, muy importante para saber la disposición final de los componentes, es una aproximación a cómo ha de quedar la tarjeta después de soldar, para realizarlo se debe escoger la opción **Output** y **3D Visualization**.

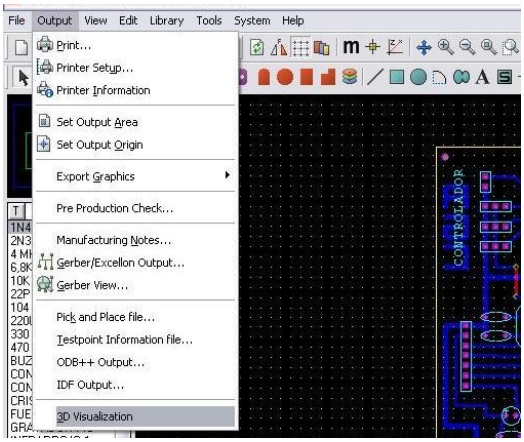


Figura 61. Menú Output opción 3D Visualization.

Fuente: Goyeneche, C. (2016) Datos de estudio.

La figura 62, muestra la tarjeta PCB en 3D con los componentes.

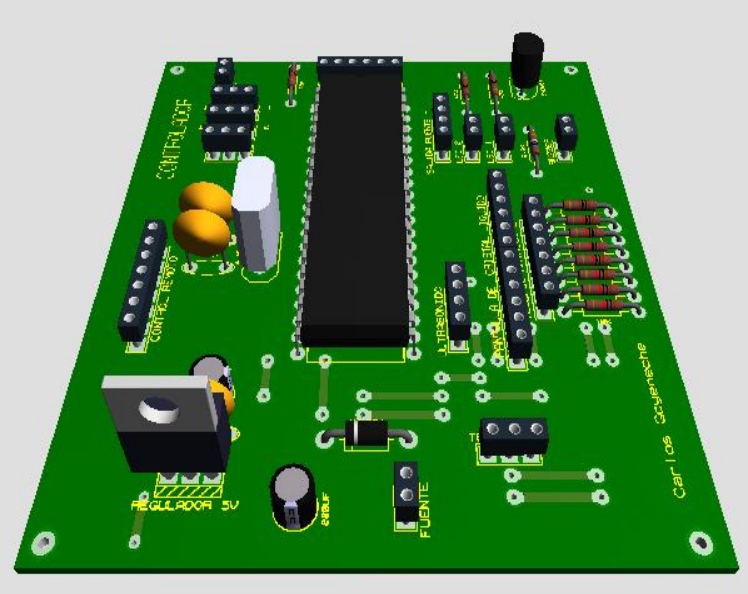


Figura 62. Tarjeta PCB en 3D con ARES.

Fuente: Goyeneche, C. (2016) Datos de estudio.

3.2.4. Construcción de la tarjeta PCB

La construcción de las tarjetas impresas o PCB de este proyecto se realizan con el método de transferencia de calor, esta es la forma más económica e ideal, con buenos resultados, los materiales básicos necesarios son:

- Una placa de cobre con baquelita conocida como pertinax, o se puede usar la placa de cobre con fibra de vidrio que tiene mejores propiedades mecánicas y además es más bonita.
- Se requiere cloruro férrico, que es el ácido para disolver el cobre, se consigue en forma granulada.
- Para la transferencia térmica del tóner se usa una plancha de ropa.
- Un recipiente plástico adecuado para el tamaño de la tarjeta. Papel fotográfico o acetato para realizar la impresión del diseño de la tarjeta.

- Lija fina o esponjilla de brillo, para limpiar la superficie de cobre.
- Marcador de tinta indeleble, útil para reparar errores.
- Por ultimo una impresora láser o fotocopiadora, para realizar la impresión del diseño previamente realizado en Ares de Proteus.

Proceso de grabado

Luego de haber realizado el diseño del circuito en Ares, se realiza la impresión en láser con una resolución máxima, para que quede con un buen volumen de tóner en el acetato. Se debe tener cuidado con la impresión del circuito para que salga por el lado correcto, a escala real y en modo espejo de la placa diseñada.

Con el diseño ya impreso en el acetato, luego procedemos a preparar la placa de cobre. Para la mejor transferencia del tóner, se debe pulir con la lija fina o con la esponjilla de brillo, se debe limpiar muy bien ya que el polvillo es perjudicial porque impide la transferencia del tóner, al igual no se debe tocar la placa con los dedos para no dejar en ella grasa, se puede limpiar con alcohol.

La transferencia del circuito a la placa de cobre se realiza posicionando el acetato encima de la placa de cobre, cara abajo o mejor con la impresión frente al cobre, se asegura con cinta adhesiva, la placa se pone en una superficie que soporte el calor por ejemplo una baldosa, seguido se pone una hoja de papel sobre el acetato, luego se enciende la plancha lo más caliente posible, se plancha el papel sobre la placa de cobre abarcando toda la placa con ligera presión, pasando la plancha del centro a los lados, es en este momento donde el tóner se transfiere a la placa.

Una vez terminado este trabajo se debe despegar el acetato de la placa de cobre, puesto que gracias al calor tiende a adherirse al cobre. Se logra un buen resultado usando agua fría, se inicia despegando una esquina solo un poco, se deja entrar agua y esta realiza el resto del trabajo, además ayuda a que el tóner no se suelte fácil.

Ya con el circuito transferido a la placa de cobre, lo que sigue es calentar agua hasta hervir, se vierte en el recipiente plástico y se le agrega el cloruro férrico, es ahora cuando se introduce la placa de cobre grabada, para que el proceso de degradado sea rápido se agita el recipiente plástico sin regar la sustancia, pues esta mancha y es muy difícil de remover, puede tomar algunos minutos el proceso.

Cuando se ha terminado de corroer, el cobre sobrante se lava bien la placa y se limpia de nuevo con la esponjilla. Es tiempo de realizar las perforaciones para introducir los componentes, se recomienda el uso de una broca 0,8 milímetros para la mayoría de los componentes pequeños y una de 1,0 milímetros para los demás pines.

Es de recordar que no es bueno realizar PCB con pistas por debajo del grosor de 0,6 milímetros, ya que no es posible realizar soldaduras, además para que el cobre no se oxide, después de soldar los componentes, conviene pintar la placa con algún esmalte que no conduzca electricidad, por ejemplo barniz dieléctrico.

Terminado el proceso la tarjeta PCB queda lista para realizar el montaje de los componentes, figura 63.

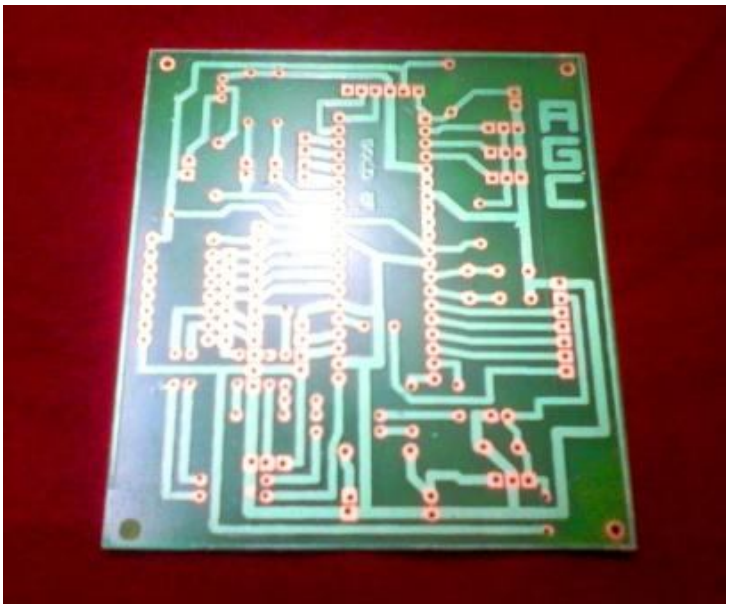


Figura 63. Fotografía tarjeta PCB terminada.

Fuente: Goyeneche, C. (2016) Datos de estudio.

3.2.5. Técnica para soldar y fijar los componentes

Herramientas:

El cautín, es la herramienta que calienta el estaño hasta derretirlo, debe ser su punta lo más delgado posible, no se debe raspar para no dañar el revestimiento de plata o como consecuencia pierde adherencia del estaño; lo mejor es limpiar con una espuma mojada.

Estaño, es un metal maleable que se funde fácilmente, se recomienda comprar un rollo de media libra o por metros que sea delgado con una mezcla en proporción de 30 de plomo y 70 de estaño.

Pasta para soldar, se usa como catalizador fundente y es muy necesaria para que el estaño se difumine mejor. Se recomienda antes de soldar aplicar pasta de soldar, pues esto ayuda a la transferencia de calor.

- Lo primero es introducir el pin del componente por el orificio de la placa PCB y sujetar el componente en su lugar evitando que se mueva al momento de soldar.
- Con la punta del cautín tocar justo en el lugar donde se desea estañar, conviene calentar el pin del componente y que este transfiera el calor a la pista.
- Cuando este caliente el pin y la superficie de la placa, se aplica estaño de manera que forme una especie de cono sin separar la punta del cautín. Se debe mantener la punta del cautín para que la soldadura se distribuya uniformemente.
- Mantener el componente inmóvil durante unos instantes para que se enfríe y se solidifique el estaño, no recomiendo soplar porque se reduce la resistencia mecánica de la soldadura.
- Recortar el pin sobrante, procurando que el corte sea lo más estético posible, lo último es limpiar bien la placa PCB con thinner, usando un cepillo de dientes.

La figura 64, muestra una fotografía de la tarjeta PCB con los componentes instalados y lista para montar los cables. Además se ve una pequeña tarjeta sobresaliente encerrada en un óvalo de color azul, que corresponde al receptor RF M40-RFC, que es la encargada de recibir por radio frecuencia los comandos del control remoto.

Figura 64. Fotografía de la tarjeta PCB con componentes montados.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Las conexiones hacia los demás circuitos, se realizan por medio de cables con terminales de pin hembra, de diferentes colores para fácil identificación, muy usados en las plataformas de desarrollo Arduino; sirven para modificar la posición de los cables si es necesario o si se requiere retirar la tarjeta.

Con la tarjeta PCB de control terminada, se realiza el montaje de la misma en la plataforma del prototipo robot. Para lograr este objetivo solo es necesario hacer cuatro orificios en las esquinas de la tarjeta PCB y en la pieza **A1** correspondiendo a las dimensiones, luego usando tornillos de rosca fina y tuercas se fija la tarjeta.

La mayoría de las veces se usan por cada tornillo dos o tres tuercas, en contratuerca, para dar estabilidad entre las piezas a unir y evitar que se aflojen.

La figura 65, muestra la fotografía de la tarjeta PCB controladora ya instalada en la plataforma, se ven los cables que conectan los circuitos.

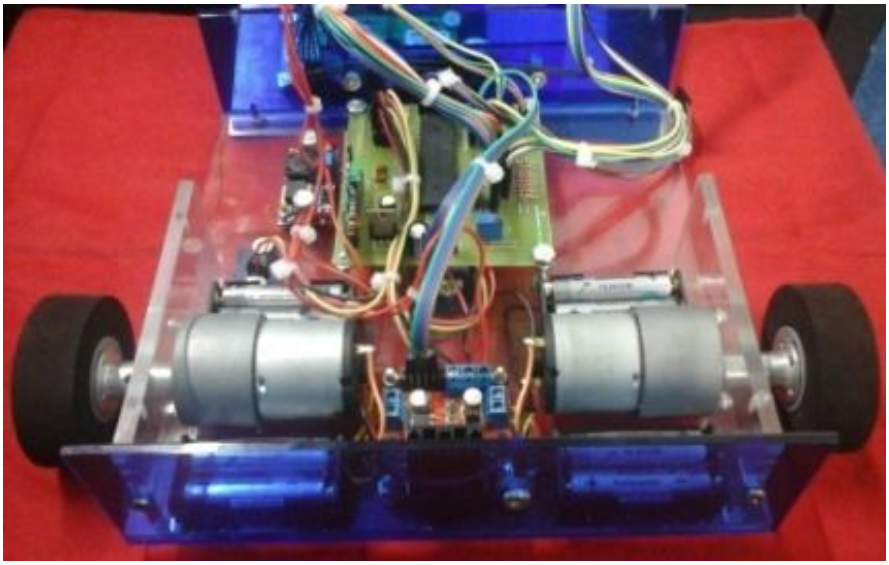


Figura 65. Fotografía de la tarjeta PCB montada en la plataforma.

Fuente: Goyeneche, C. (2016) Datos de estudio.

Se usan amarres plásticos, que sirven para acomodar los cables de forma que se vean ordenados y permiten que no se suelten de los pines en las tarjetas del prototipo. En los terminales de los cables se les aplica silicona en barra, calentándola para que se adhiera, de manera que sujeten los mismos y estos no se suelten.

3.3. PROGRAMACIÓN DEL PROTOTIPO ROBOT

Se desarrollan dos programas en asembler, para probar la cinemática del prototipo robot y ver el funcionamiento de los circuitos electrónicos. Se decide experimentar la programación en Mplab, para realizar un seguimiento a nivel máquina del microcontrolador.

3.3.1. Pruebas con MPLAB

Primer algoritmo. Diseñar una estrategia y fijar el algoritmo de control, luego escribir el programa, una vez compilado cargarlo en el microcontrolador del simulador. Dependiendo de la posición de los sensores con respecto a la línea se programa para que el prototipo robot tome una decisión.

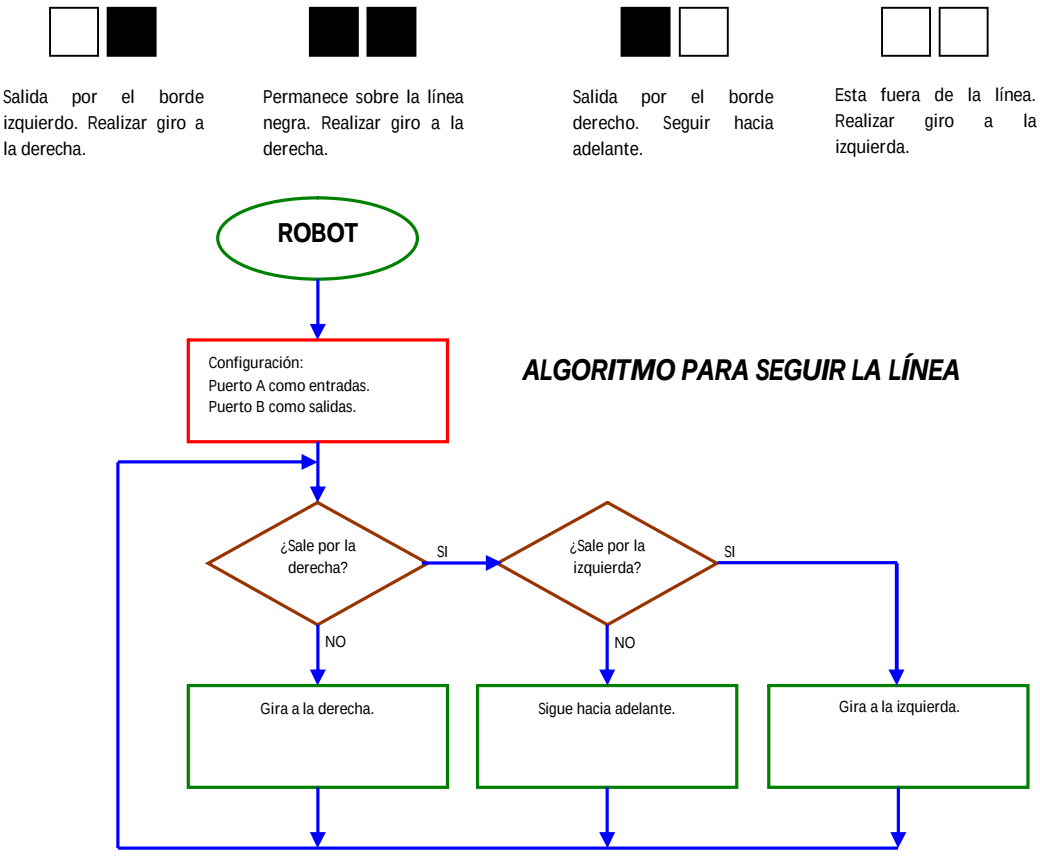


Figura 66. Diagrama de decisiones para el primer algoritmo.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La figura 66, muestra la representación de los sensores sobre la línea, sirve para entender mejor el planteamiento del algoritmo, se indican las posibles combinaciones de entrada y de este análisis se obtiene el diagrama de flujo.

El programa se escribe sobre el software Mplab, este es un editor IDE gratuito, destinado a productos de la marca Microchip, se puede seleccionar los distintos microcontroladores, además de permitir la grabación de estos circuitos integrados directamente al grabador.

Mplab, es un programa que corre bajo Windows y presenta las clásicas barras de programa, de menú, de herramientas de estado, entre otras. Posee editor de texto, compilador y simulación, es de libre distribución y se puede descargar de la página de Internet http://www.microchip.com/ de Microchip, Enrique Palacios (2008).

Luego de instalar el programa en la computadora, aparece el icono en el escritorio como lo muestra la figura 67.

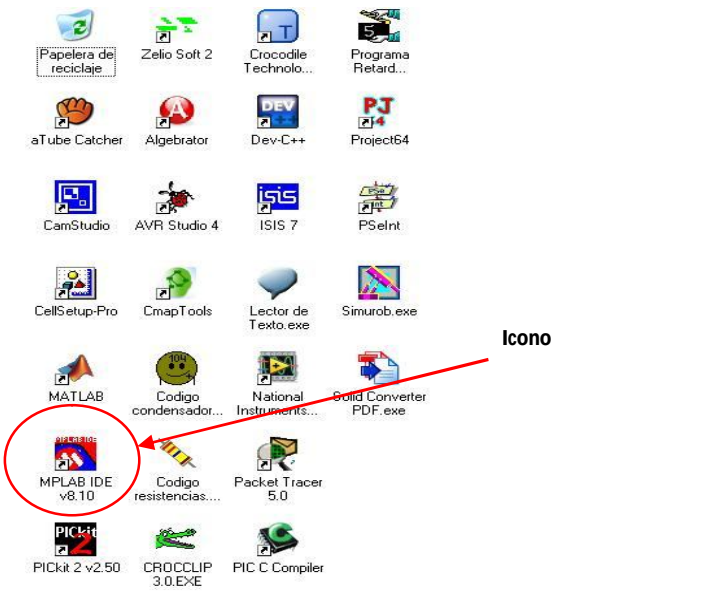


Figura 67. Icono Mplab en escritorio.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Al dar doble clip en el ícono se inicia el programa; su entorno gráfico es mostrado en la figura 68, donde se aprecia los menús de archivos, edición, vistas, proyectos, compilador, programación, herramientas, configuración, pantallas y ayudas.

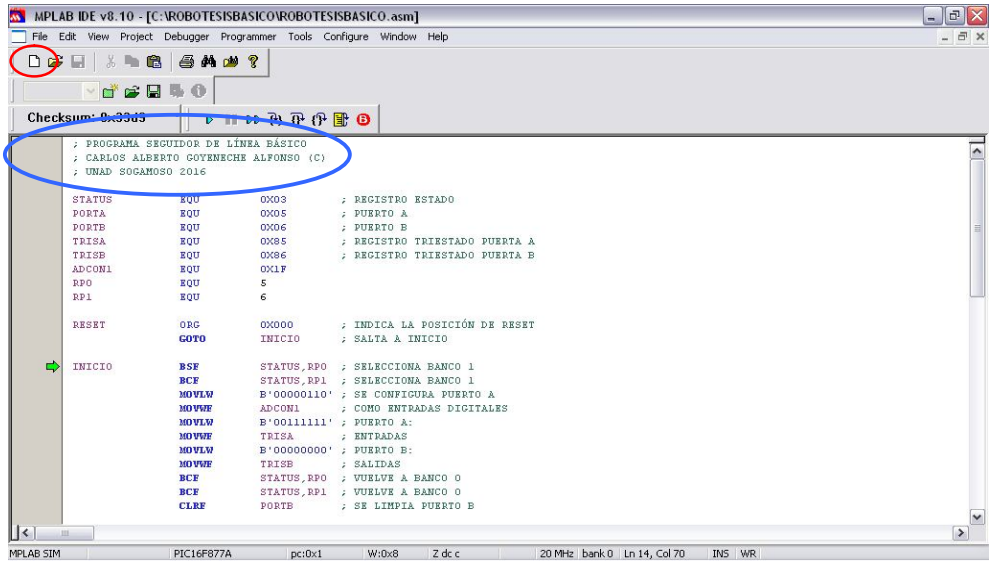


Figura 68. Ventana de trabajo de Mplab.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Para escribir un programa, se da clip en el ícono similar a una hoja en blanco (óvalo rojo) “New file” y aparece una ventana marcada como *sin título* (untitled), en esta nueva ventana es donde se escriben los comentarios, el código y la sintaxis del programa.

Luego de haber escrito algo de programa, se pasa a dar clip sobre el ícono de guardar el archivo, este toma por defecto la extensión .mcw, pero es necesario realizar simulación, modificar el código, verificar errores, aquí se debe guardar el programa como .asm; además sobre este último es que trabaja el compilador, se ubica la carpeta donde se ha de guardar el proyecto, se cambia la extensión .mcw por .asm, se nombra el programa y se guarda.

Para seleccionar el microcontrolador a usar, se realiza dando clip en “Configure”, luego en seleccionar equipo “Select device” y aparece una nueva ventana. Al desplegar la lista de componentes se busca, el componente a usar y se marca; para este proyecto el PIC16F877A, seguido se da OK. En el menú configuración se escoge el tipo de reloj a usar, para nuestro caso es el XT (alta velocidad), para un cristal de 4 mega Hertz.

Ahora se puede escribir el algoritmo anterior; este primer programa se nombra **ROBOTESISBASICO.asm**.

Introducción del programa: Inicialmente se realiza una presentación personalizada del programa, estas líneas no son tomadas por el compilador, ya que están precedidas del punto y coma, se pueden usar los que uno quiera, porque estos no se convierten en **.hex**, y no ocupan memoria en el microcontrolador.

```
; PROGRAMA SEGUIDOR DE LÍNEA BÁSICO
; CARLOS ALBERTO GOYENECHE ALFONSO (C)
; UNAD SOGAMOSO 2016
```

Asignación de memoria: El siguiente bloque de instrucciones corresponde a un método de asignación de etiquetas, donde se asigna usando la instrucción **EQU**, van al principio del programa y antes de las instrucciones. Siempre que aparezca el nombre en el programa este será sustituido por su valor numérico que se le haya asignado. Generalmente la etiqueta en su nombre se describe el valor de manera significativa, además se usa para definir constantes y direcciones en la memoria; deben ser fáciles de recordar y tener afinidad para el diseñador del programa.

```
STATUS      EQU     0X03        ; REGISTRO DE ESTADO
PORTA       EQU     0X05        ; PUERTO A
PORTB       EQU     0X06        ; PUERTO B
TRISA       EQU     0X85        ; REGISTRO TRI ESTADO PUERTA A
TRISB       EQU     0X86        ; REGISTRO TRI ESTADO PUERTA B
ADCON1      EQU     0X1F        ; SELECCIÓN DE TRABAJO DEL PUERTO
RP0         EQU     5
RP1         EQU     6
```

Reinicio: Se asigna el punto de inicio luego de un reset.

```
RESET       ORG     0X000       ; INDICA LA POSICIÓN DE RESET
            GOTO    INICIO      ; SALTA A INICIO
```

Configuración de puertos: Se definen los pines del PORTA como entradas, se carga el registro TRISA con el valor B'00111111' y los pines del PORTB se usan todos como salidas, se carga TRISB con el valor B'00000000', pero antes se debe haber ingresado al banco 1 en el registro STATUS, modificando RP0 y RP1. Es en el banco 1 donde se encuentran los registros de configuración, además se debe

tener en cuenta de establecer el puerto A como entradas digitales en el registro ADCON1 o este puerto tendrá lecturas erróneas.

```
INICIO      BSF     STATUS,RP0      ; SELECCIONA BANCO 1
            BCF     STATUS,RP1      ; SELECCIONA BANCO 1
            MOVLW   B'00000110'     ; SE CONFIGURA PUERTO A
            MOVWF   ADCON1          ; COMO ENTRADAS DIGITALES
            MOVLW   B'00111111'     ; PUERTO A:
            MOVWF   TRISA           ; ENTRADAS
            MOVLW   B'00000000'     ; PUERTO B:
            MOVWF   TRISB           ; SALIDAS
            BCF     STATUS,RP0      ; VUELVE A BANCO 0
            BCF     STATUS,RP1      ; VUELVE A BANCO 0
            CLRF    PORTB           ; SE LIMPIA PUERTO B
```

Rutina principal: Después se realiza el programa principal según lo planteado en el diagrama de flujo de la figura 66:

```
PRINCIPAL   MOVLW   B'00001110'     ; DATO PARA GIRAR A LA DERECHA.
            BTFSS   PORTA,1         ; ¿HA SALIDO POR LA DERECHA?
            GOTO    SALIDA          ; NO, GIRA A LA DERECHA.
            MOVLW   B'00000001'     ; DATO PARA GIRAR A LA IZQUIERDA.
            BTFSS   PORTA,0         ; ¿HA SALIDO POR LA IZQUIERDA?
            MOVLW   B'00001000'     ; NO, SIGUE RECTO.

SALIDA      MOVWF   PORTB           ; MUEVE EL DATO DE W A PUERTO B.
            GOTO    PRINCIPAL       ; VUELVE A CORRER CICLO.


            END
```

Fin: Por último se debe indicar el fin del programa, **END** directiva obligatoria, ya que el programa ensamblador debe saber dónde detener el proceso, es por ello que es la última línea del programa.

Una vez escrito el programa de pruebas básico, lo que sigue es compilarlo para ver si hay errores de sintaxis, es importante tener en cuenta que el compilador no advierte de errores lógicos o de estructura del programa. La compilación se realiza dando clic en “Project” en la barra de herramientas, luego en “Quickbuild ROBOTESISBASICO.asm”, el software realiza un escaneo y compila el código escrito y da forma al .hex e indica si hay errores, de no encontrar los mismos el programa se puede simular. Para simular en Mplab el programa se debe dar clic en “Debugger”, luego en “Select Tool” y por último en “MPLAB SIM”.

Aparecen los botones de la figura 69.

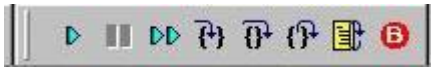


Figura 69. Barra de botones para simulación de Mplab.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Después se escoge en el menú “View”, dando clic la opción “ Special Function Registers” y aparece una ventana como la vista en la figura 70, ordenada en la opción “Window” y vertical “Tile Vertically”, de manera que se puedan ver las dos ventanas al tiempo, la de escritura del programa y la de los registros de funciones especiales.

.

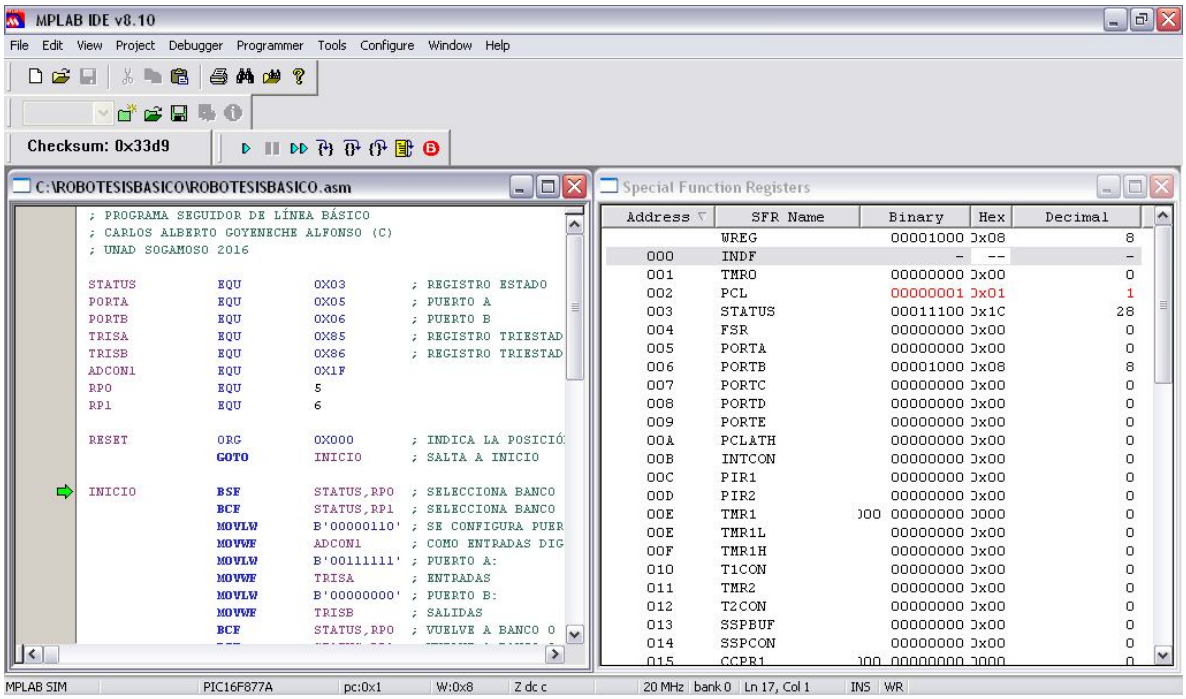


Figura 70. Escritorio de Mplab para simulación.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Para ver el programa paso a paso solo es necesario oprimir la tecla F7 y para reiniciar el programa se oprime la tecla F6, de esta manera se comprueba si hace lo esperado, antes de montarlo en el simulador de Proteus o en el microcontrolador.

Para realizar la simulación en Proteus, se procede a dar doble clip en el archivo creado nombre de CONTROLADOR.DSN, al realizarlo aparece la ventana que se muestra en la figura 71. Se puede apreciar el esquema electrónico del proyecto.

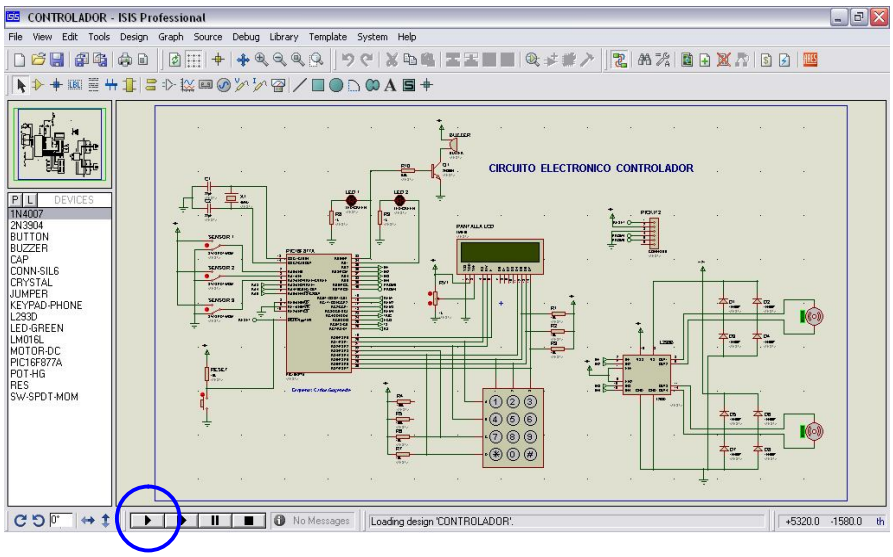


Figura 71. Ventana de entorno en Proteus ISIS para simulación.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Para cargar en el microcontrolador lo compilado en archivo .hex, solo es necesario buscar el símbolo del mismo y dar doble clip encima, aqui aparece la siguiente ventana, figura 72.

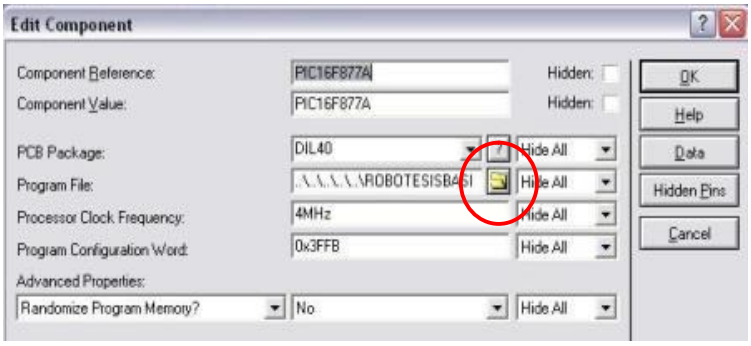


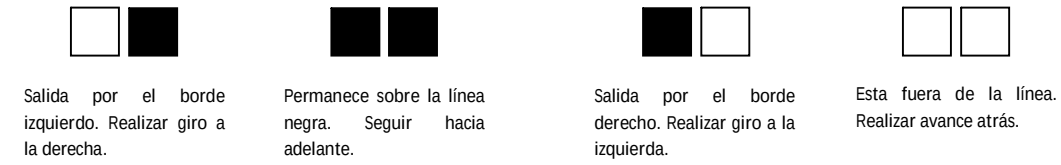
Figura 72. Ventana de propiedades del microcontrolador en ISIS.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Se debe dar clip en el icono de la carpeta, ovalo rojo en la figura 72 y buscar la ubicación del archivo .hex. Al ubicarlo se da doble clip en el archivo y por último se da OK para guardar. Al dar clip en el icono play del entorno en Proteus se inicia la simulación, ovalo azul de la figura 71.

Al segundo programa se nombra **ROBOTESISMEJORADO.asm**.

Segundo algoritmo. Dependiendo de la posición de los sensores con respecto a la línea se programa para que el robot tome una decisión, figura 73.



Salida por el borde izquierdo. Realizar giro a la derecha.

Permanece sobre la línea negra. Seguir hacia adelante.

Salida por el borde derecho. Realizar giro a la izquierda.

Esta fuera de la línea. Realizar avance atrás.

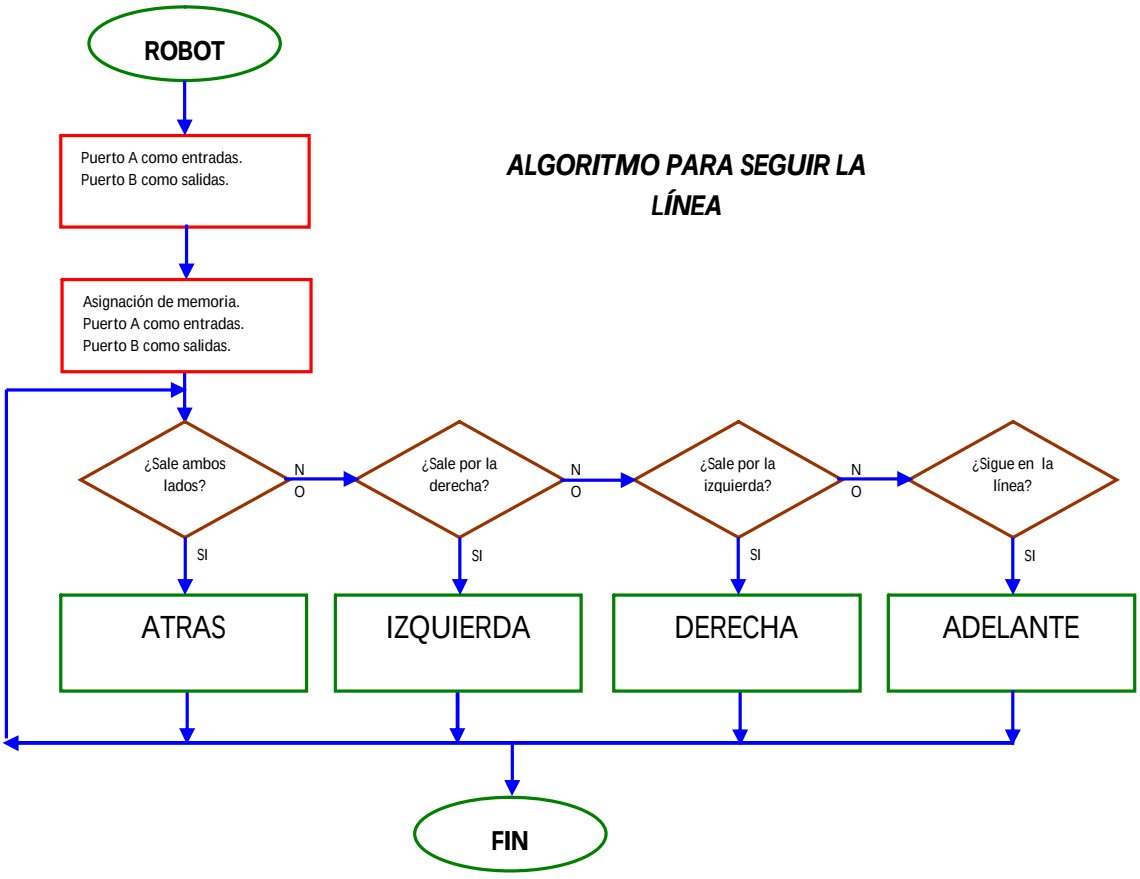


Figura 73. Diagrama de decisiones para segundo algoritmo.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Introducción del programa: Se realiza la presentación personalizada del programa, como se dijo antes estas líneas no son tomadas por el compilador ya que están precedidas del punto y coma.

```
; PROGRAMA SEGUIDOR DE LÍNEA BÁSICO
; CARLOS ALBERTO GOYENECHE ALFONSO (C)
; UNAD SOGAMOSO 2016
```

Asignación de memoria: Bloque de instrucciones con el método de asignación de etiquetas.

```
STATUS      EQU     0X03        ; REGISTRO ESTADO
PORTA       EQU     0X05        ; PUERTO A
PORTB       EQU     0X06        ; PUERTO B
TRISA       EQU     0X85        ; REGISTRO TRIESTADO PUERTA A
TRISB       EQU     0X86        ; REGISTRO TRIESTADO PUERTA B
ADCON1      EQU     0X1F        ; SELECCION DE TRABAJO DEL PUERTO
RP0         EQU     5           ; OPCIONES EN REGISTRO STATUS
RP1         EQU     6           ; PARA LA SELECCIÓN DE BANCO
AUX1        EQU     20          ; REGISTRO MEMORIA ATRAS
AUX2        EQU     21          ; REGISTRO MEMORIA IZQUIERDA
AUX3        EQU     22          ; REGISTRO MEMORIA DERECHA
AUX4        EQU     23          ; REGISTRO MEMORIA ADELANTE
TIME        EQU     24          ; REGISTRO MEMORIA CONTADOR
```

Reinicio: Se asigna el punto de inicio luego de un reset.

```
RESET       ORG     0X000       ; INDICA LA POSICIÓN DE RESET
            GOTO    INICIO      ; SALTA A INICIO
```

Rutina de espera: Se tiene un tiempo de espera, que es usado para mantener una condición ya sea para lectura o para operar una salida. Por ejemplo un disparo prolongado.

```
PAUSA       MOVLW   D'150'      ; RUTINA QUE SIRVE PARA
            MOVWF   TIME        ; MANEJAR LOS TIEMPOS
            NOP                 ; DE DISPARO DE SALIDA
TRON        DECFSZ  TIME        ; A LOS MOTORES, DONDE
            GOTO    TRON        ; SE DECREMENTA UN VALOR
            NOP                 ; ALMACENADO EN EL REGISTRO
            RETLW   00          ; TIEMPO
```

Configuración de puertos: Se definen los pines del PORTA como entradas y los pines del PORTB se usan todos como salidas, antes se debe haber ingresado al banco 1 en el registro STATUS, modificando RP0 y RP1. Es en el banco 1 donde

se encuentran los registros de configuración, además se debe tener en cuenta de establecer el puerto A como entradas digitales en el registro ADCON1 o este puerto tendrá lecturas erróneas.

```
INICIO      BSF     STATUS,RP0      ; SELECCIONA BANCO 1
            BCF     STATUS,RP1      ; SELECCIONA BANCO 1
            MOVLW   B'00000110'     ; SE CONFIGURA PUERTO A
            MOVWF   ADCON1          ; COMO ENTRADAS DIGITALES
            MOVLW   B'00111111'     ; PUERTO A:
            MOVWF   TRISA           ; ENTRADAS
            MOVLW   B'00000000'     ; PUERTO B:
            MOVWF   TRISB           ; SALIDAS
            BCF     STATUS,RP0      ; VUELVE A BANCO 0
            BCF     STATUS,RP1      ; VUELVE A BANCO 0
            CLRF    PORTB           ; SE LIMPIA PUERTO B
```

Rutina de carga memoria: Se realiza la rutina de cargar datos en los registros, para que sirvan de memoria.

```
            MOVLW   B'00000000'     ; RUTINA PARA CARGAR
            MOVWF   AUX1            ; LOS DATOS EN LAS
            MOVLW   B'00000010'     ; MEMORIAS DE COMPARACIÓN
            MOVWF   AUX2            ; Y LUEGO ESTOS DATOS SE
            MOVLW   B'00000001'     ; USAN PARA SER COMPARADOS
            MOVWF   AUX3            ; CON LOS DATOS LEIDOS
            MOVLW   B'00000011'     ; POR EL PUERTO A DEL
            MOVWF   AUX4            ; MICROCONTROLADOR
```

Rutina principal: La rutina principal del programa se basa en un ciclo infinito, tiene en particular la comparación de un dato almacenado en registros de memoria y el dato leído en el puerto A. Al realizar esta operación se obtiene como resultado dos posibilidades: es igual y la bandera del registro STATUS se establece en uno o no es igual y la bandera del registro STATUS se establece en cero.

```
CICLO       MOVF    PORTA,0         ; LEE TODO EL PUERTO A, Y LO
            XORWF   AUX1,0          ; COMPARA CON EL DATO ALMACENADO
            BTFSC   STATUS,2        ; EN MEMORIA, SI EL RESULTADO ES
            GOTO    ATRAS           ; UNO SALTA A RUTINA ATRÁS.

            MOVF    PORTA,0         ; LEE TODO EL PUERTO A, Y LO
            XORWF   AUX2,0          ; COMPARA CON EL DATO ALMACENADO
            BTFSC   STATUS,2        ; EN MEMORIA, SI EL RESULTADO ES
            GOTO    IZQUIERDA       ; UNO SALTA A RUTINA IZQUIERDA

            MOVF    PORTA,0         ; LEE TODO EL PUERTO A, Y LO
            XORWF   AUX3,0          ; COMPARA CON EL DATO ALMACENADO
            BTFSC   STATUS,2        ; EN MEMORIA, SI EL RESULTADO ES
            GOTO    DERECHA         ; UNO SALTA A RUTINA DERECHA

            MOVF    PORTA,0         ; LEE TODO EL PUERTO A, Y LO
            XORWF   AUX4,0          ; COMPARA CON EL DATO ALMACENADO
            BTFSC   STATUS,2        ; EN MEMORIA, SI EL RESULTADO ES
            GOTO    ADELANTE        ; UNO SALTA A RUTINA ADELANTE

            GOTO    CICLO           ; VUELVE A INICIAR LA RUTINA
```

Rutinas de respuestas: La rutina principal del programa procesa los datos obtenidos por la lectura del puerto y debe tomar una decisión como respuesta a la salida. Es por esto que se tienen pequeñas rutinas de salida.

```
ATRAS       MOVLW  B'00011011'    ; MUEVE DATO AL PUERTO B
            MOVWF  PORTB          ; PARA QUE CONTROLE EL
            CALL   PAUSA          ; ACTUAR DE LOS MOTORES
            MOVLW  B'00000000'    ; Y ESPERA UN TIEMPO AL
            MOVWF  PORTB          ; SALTAR A PAUSA, LUEGO
            CALL   PAUSA          ; APAGA EL PUERTO Y
            GOTO   CICLO          ; VUELVE A LA RUTINA CICLO

DERECHA     MOVLW  B'00100000'    ; MUEVE DATO AL PUERTO B
            MOVWF  PORTB          ; PARA QUE CONTROLE EL
            CALL   PAUSA          ; ACTUAR DE LOS MOTORES
            MOVLW  B'00000000'    ; Y ESPERA UN TIEMPO AL
            MOVWF  PORTB          ; SALTAR A PAUSA, LUEGO
            CALL   PAUSA          ; APAGA EL PUERTO Y
            GOTO   CICLO          ; VUELVE A LA RUTINA CICLO

IZQUIERDA   MOVLW  B'00000100'    ; MUEVE DATO AL PUERTO B
            MOVWF  PORTB          ; PARA QUE CONTROLE EL
            CALL   PAUSA          ; ACTUAR DE LOS MOTORES
            MOVLW  B'00000000'    ; Y ESPERA UN TIEMPO AL
            MOVWF  PORTB          ; SALTAR A PAUSA, LUEGO
            CALL   PAUSA          ; APAGA EL PUERTO Y
            GOTO   CICLO          ; VUELVE A LA RUTINA CICLO

ADELANTE    MOVLW  B'00100110'    ; MUEVE DATO AL PUERTO B
            MOVWF  PORTB          ; PARA QUE CONTROLE EL
            CALL   PAUSA          ; ACTUAR DE LOS MOTORES
            MOVLW  B'00000000'    ; Y ESPERA UN TIEMPO AL
            MOVWF  PORTB          ; SALTAR A PAUSA, LUEGO
            CALL   PAUSA          ; APAGA EL PUERTO Y
            GOTO   CICLO          ; VUELVE A LA RUTINA CICLO

            END                   ; FIN DEL PROGRAMA.
```

Conclusión de la simulación en Proteus y prototipo

En la simulación de Proteus, se comprueba que los dos programas corren muy bien, se ve al simular el primer programa que los motores giran más rápido, debido a que los tiempos de activación son más largos podría decirse casi continuos, en cambio en la simulación del segundo programa se ven los motores girar bastante lento, esto debido a que se genera un tren de pulsos y el simulador no muestra una representación en tiempo real de los acontecimientos.

Estos dos programas en asembler, son solo para verificar la cinemática del prototipo, ya que al planearse el circuito electrónico, se incluyeron varios periféricos y se estiman varias funciones adicionales, que hacen al proyecto mucho más complejo. Luego se graban en el prototipo robot y se verifica un buen funcionamiento.

3.3.2. Programación en CCS

Para realizar la programación del microcontrolador en un lenguaje como el C, se debe utilizar un compilador de C que genera ficheros en formato hexadedimal, que es el necesario para programar un microcontrolador. El compilador de C que vamos a utilizar es el C de CCS, que nos va a permitir desarrollar las fases que componen el proyecto, desde la edición hasta la compilación pasando por la depuración de errores, García Breijo (2008).

Al igual que el compilador de Turbo C, éste "*traduce*" el código C del archivo fuente .C a lenguaje máquina para los microcontroladores PIC, generando así un archivo en formato hexadecimal .hex.

CCS fue diseñado específicamente para microcontroladores PIC, con amplia librería de funciones predefinidas, comandos de preprocesado y ejemplos, maneja controladores para diversos dispositivos como: LCD, ADC, EEPROM, entre otros y permite que al llevar a cabo un proyecto sea más fácil su desarrollo. Es por esta razón que se ha determinado el uso de este software, evitando el desarrollo de los tediosos programas en lenguaje ensamblador, antes explicado.

Luego de instalar el programa en la computadora, aparece el icono en el escritorio como lo muestra la figura 74.

Figura 74. Icono PIC C en escritorio.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Al dar doble clip en el icono se inicia el programa, su entorno gráfico es mostrado en la figura 75.

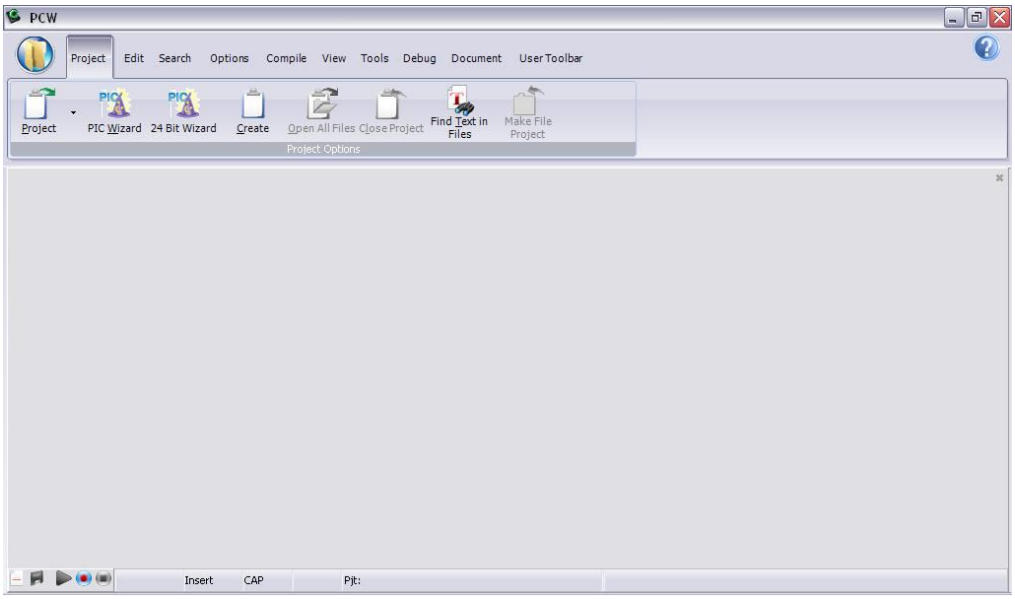


Figura 75. Ventana de entorno CCS PCW.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Después de iniciar el CCS se debe ir al asistente para la creación de proyectos o PIC Wizard, su icono se ve en la figura 76.

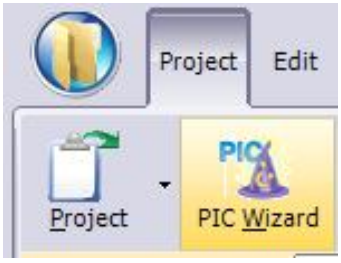


Figura 76. Icono de PIC Wizard.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Este asistente permite al usuario especificar los parámetros del proyecto, genera los archivos .c, .h y .pjt. Se da doble clip en este icono y aparece la ventana donde se guarda el proyecto, figura 77.

En esta ventana es donde se busca la carpeta que guardará el programa y sus componentes, y en esta misma ventana se da nombre al proyecto, en este caso se nombra CONTROLADOR.pjt.

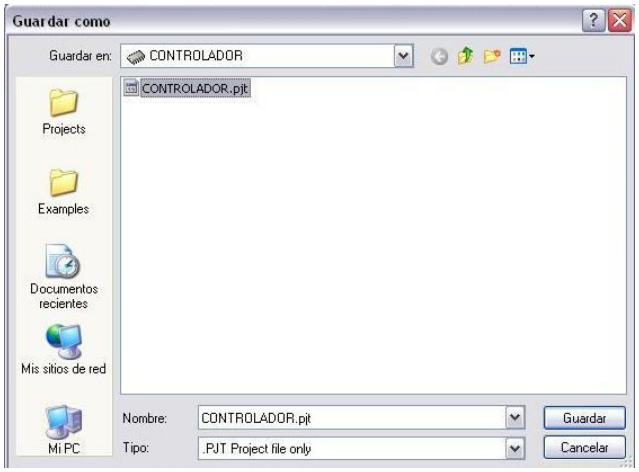


Figura 77. Ventana para guardar el PIC Wizard.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Al guardar el proyecto, aparece la siguiente ventana, figura 78. En ella se ve la configuración principal con diferentes opciones que permiten seleccionar los parámetros del proyecto, es aquí donde se define el microcontrolador a usar, el tipo de oscilador a utilizar y los fusibles necesarios. Para el proyecto se usa el PIC16F877A, con un cristal oscilador de 4MHz.

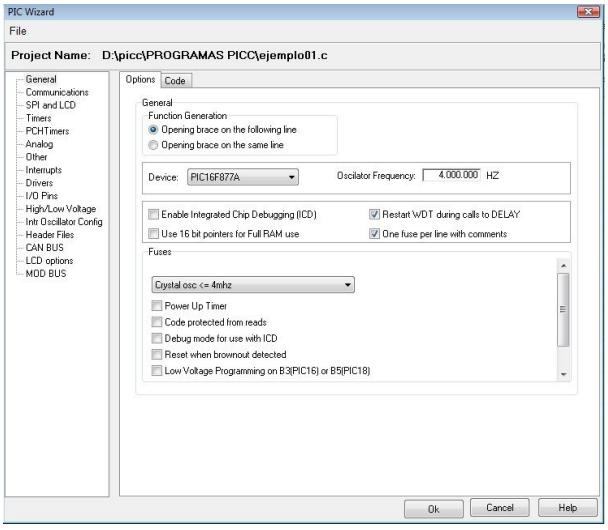


Figura 78. Ventana general PIC Wizard.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La ventana configuración de comunicaciones, figura 79. Cuenta con el protocolo RS232 serial. En esta ventana se puede configurar la comunicación entre microcontroladores PIC, la comunicación con una computadora, se puede implementar comunicación sincrónica o asincrónica. La técnica de comunicación I2C, configuración con periféricos como EEPROM, sensores y LCD, entre otros. No se usa comunicación serial en este proyecto, pero es tenida en cuenta debido a su alta importancia.

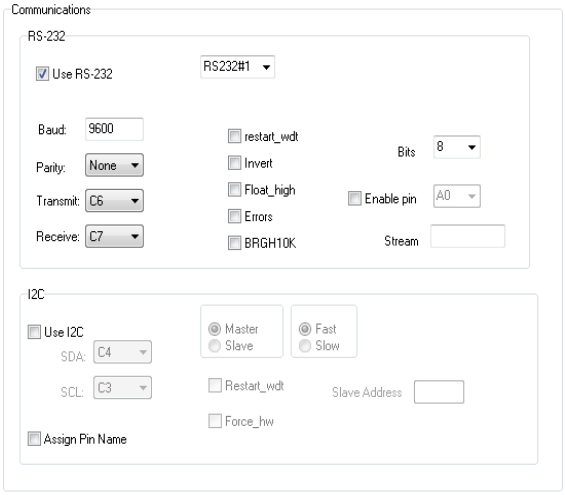


Figura 79. Ventana de comunicaciones.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

En la ventana de configuración del convertidor análogo a digital, figura 80, se pueden configurar los pines por donde se recibirán las señales análogas, los voltajes de referencia y múltiples canales de entrada. Donde se escoge la opción A0 A1 A3 con unidades de conversión 0-255.

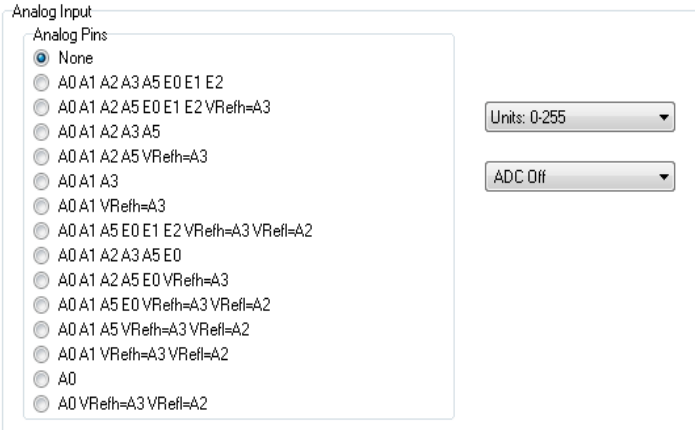


Figura 80. Ventana de entradas análogas.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La siguiente figura 81, muestra la ventana Other. En ella se puede configurar la obtención de un tiempo cuando se produce un evento en un pin del microcontrolador. Se puede comparar el valor de un temporizador con el valor de un registro y que provoque una acción determinada. Tiene la opción PWM o modulación de ancho de pulso, muy usada para aplicaciones de control de velocidad de motores y control de servomotores. No usado en este proyecto, pero comentada, ya que se puede aplicar para el manejo del puente H.

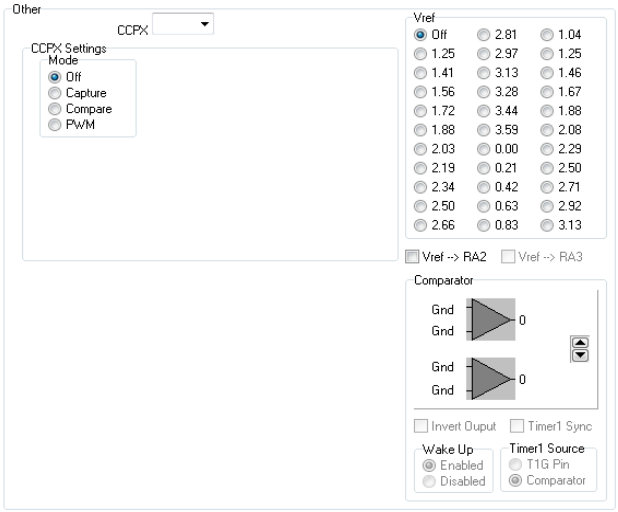


Figura 81. Ventana ara otros recursos.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La ventana para configurar interrupciones, figura 82. Se usa para que con determinado evento, por ejemplo escritura de una memoria EEPROM, cambio de estado de un pin, finalización de una conversión análoga a digital, el microcontrolador interrumpa el programa principal y ejecute una rutina de otro programa. Por ahora no usado en este proyecto.

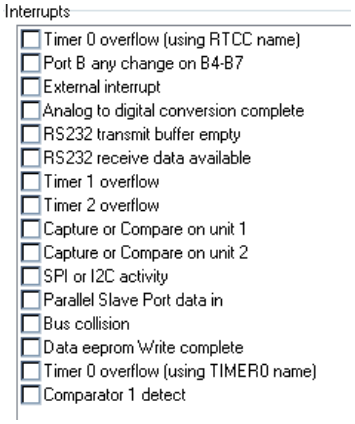


Figura 82. Ventana Interrupciones.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Ventana drivers, figura 83. Son rutinas pre programadas de dispositivos periféricos externos. Incluye convecino análogo a digital o ADC, manejo de pantallas LCD, manejo de teclados matriciales, manejo de memorias EEPROM, escritura y lectura de memorias RAM, entre otras más aplicaciones. Se usan las opciones LCD Driver y 3X4 Keypad driver.

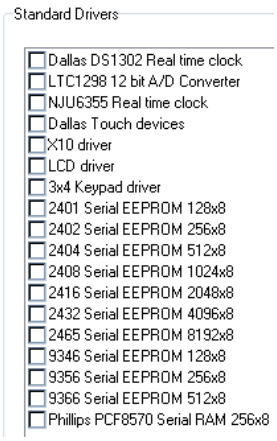


Figura 83. Ventana Drivers.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La ventana de configuración de pines del microcontrolador, figura 84. Se usa para asignar a los pines del microcontrolador funciones de entrada o salida, algunos pines pueden ser configurados como análogos, además se pueden habilitar resistencias Pullups para algunas aplicaciones en el puerto B.

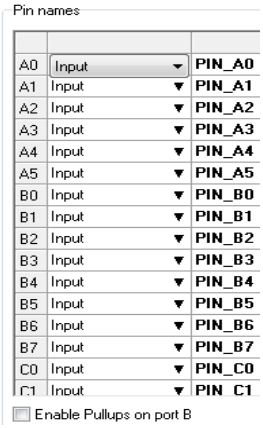


Figura 84. Ventana configuración de pines.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Una vez terminada la configuración necesaria para el proyecto, se deben guardar los cambios dando clip en **OK**.

Rutinas de presentación y clave de acceso.

Inicia con la definición de las directivas, que son las encargadas de controlar la conversión del programa a código máquina por parte del compilador.

```
#include <CONTROLADOR.h>          //Incluir CONTROLADOR .hex.
#include <LCD.C>                  //Incluir librería LCD.
#include <KBD.C>                  //Incluir librería Teclado de 3X4.
```

A continuación se ve la función principal, en donde se definen las variables y se ajustan las funciones a realizar por los puertos del microcontrolador.

```
void main()
{                                 //Programa principal.

  setup_adc_ports(AN0_AN1_AN3);   //Se usan los pines A0, A1 y A3 como entradas análogas.
  lcd_init();                     //Se inicializa librería Pantalla LCD.
  kbd_init();                     //Se inicializa librería Teclado matricial.

        char a='4', b='3', c='7';       //Clave de acceso 437.
        char a1=0, b1=0, c1=0;          //Variables que almacenan valores de la clave.
        char d;                         //Variable  d=carácter estación.
        char m;                         //Variable  m=carácter volver.
        char r1;                        //Variable  r1=carácter clave incorrecta.
        int e;                          //Variable  e=decimal estación.
        int f;                          //Variable  f=sensor derecho.
        int g;                          //Variable  g=sensor izquierdo.
        int h;                          //Variable  h=contador de estaciones.
        int i;                          //Variable  memoria de estación.
        int k;                          //Variable del contador.
        int16 d1, t1;                   //Variables de ultrasonido d1=distancia t1=tiempo.

  setup_adc(ADC_CLOCK_DIV_4);           //Se ajusta el tiempo de conversión análoga.
  setup_timer_1(T1_INTERNAL|T1_DIV_BY_8); //Se ajusta temporizador interno.
```

Siguen las instrucciones que forman el programa, indican que debe realizar el microcontrolador y para explicar que hace cada instrucción se incluyen los comentarios, precedidos de dos barras inclinadas //.

```
        lcd_putc("\f   INICIANDO   ");   //Muestra "INICIANDO".
        delay_ms(400);                   //Espera 0,4 segundos.
        lcd_putc("\n ---           ");   //Muestra 3 líneas.
        delay_ms(400);                   //Espera 0,4 segundos.
        lcd_putc("\n    ---        ");   //Muestra 3 líneas.
        delay_ms(400);                   //Espera 0,4 segundos.
        lcd_putc("\n       ---     ");   //Muestra 3 líneas.
        delay_ms(400);                   //Espera 0,4 segundos.
        lcd_putc("\n          --- ");   //Muestra 3 líneas.
        delay_ms(400);                   //Espera 0,4 segundos.
```

La rutina anterior realiza una presentación en la pantalla LCD, donde aparece la palabra “INICIANDO” y en la segunda línea de la pantalla se ven tres líneas desplazándose de izquierda a derecha.

La siguiente rutina realiza la visualización en la pantalla LCD de los caracteres “ROBOT TRANSPORTE" y en la segunda línea "Carlos Goyeneche", dando una presentación del objeto del prototipo o nombre del mismo y el nombre de su diseñador, figura 85.

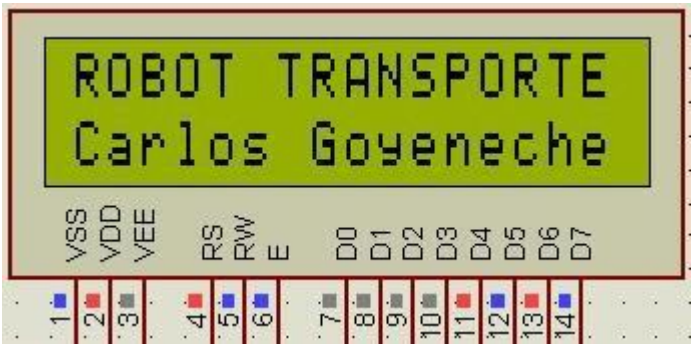


Figura 85. Presentación en la pantalla LCD de la rutina Inicial.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

```
lcd_putc("\fROBOT TRANSPORTE");     //Muestra "ROBOT TRANSPORTE".
delay_ms(2000);                     //Espera 2 segundos.
lcd_putc("\nCarlos Goyeneche");     //Muestra "Carlos Goyeneche".
delay_ms(3000);                     //Espera 3 segundos.
output_high(PIN_B1);                //Enciende los LEDs frontales
delay_ms(1000);                     //Espera 1 segundo
output_low(PIN_B1);                 //Apaga los LEDs
delay_ms(3000);                     //Espera 3 segundos.
```

Las rutinas anteriores solo se realizan cuando se energiza el prototipo o cuando se produce un reinicio externo, ya que no hacen parte del programa cíclico o de bucle infinito.

Se busca que el programa realice una aplicación de acceso con clave para las tareas, pensado en la importancia de controlar los equipos y dar asignación solo a personal autorizado. La clave de ingreso es: cuatro (4), tres (3) y siete (7), solo se puede modificar por programa en CCS. Si al digitar la clave esta es incorrecta el prototipo anuncia dando tres “bips” y no permite ingresar a trabajo, el programa le permite volver a intentar ingresar la clave las veces que desee.

La instrucción while(1), se utiliza para repetir sentencias en un bucle sin fin, esto para que no retorne a las rutinas iniciales.

```
while(1)
```

Inicializa las variables en cero.

```
{                       //Ciclo de clave de acceso.
a1=0;                   //Se asigna 0 a variable a1.
b1=0;                   //Se asigna 0 a variable b1.
c1=0;                   //Se asigna 0 a variable c1.
r1=0;                   //Se asigna 0 a variable r1.
h=0;                    //Se asigna 0 a variable h.
```

Muestra en la pantalla LCD los datos y pide al usuario introducir la clave número a número. Cada vez que se oprime una tecla suena un “bip”. Figura 86.

Figura 86. Presentación en la pantalla LCD de la rutina clave.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

```
lcd_putc("\fCLAVE DE INGRESO");     //Muestra "CLAVE DE INGRESO".
lcd_putc("\nPARA DIRECCIONAR");     //Muestra "PARA DIRECCIONAR".
delay_ms(2000);                     //Espera 2 segundos.
lcd_putc("\fINGRESE EL..");         //Muestra "INGRESE EL..".

lcd_putc("\nPRIMER NUMERO   ");     //Muestra "PRIMER NUMERO".
```

Lee primer número oprimido en el teclado y lo guarda en la variable a1.

```
Do{
        a1=kbd_getc();              //Lee teclado y asigna a la variable a1.
        }
while(a1==0);                       //
output_high(PIN_B0);                //Activa buffer.
delay_ms(200);                      //Espera 0,2 segundos.
output_low(PIN_B0);                 //Apaga buffer.
delay_ms(100);                      //Espera 0,1 segundos.

lcd_putc("\nSEGUNDO NUMERO  ");     //Muestra "SEGUNDO NUMERO".
```

Lee segundo número oprimido en el teclado y lo guarda en la variable b1.

```
Do{
        b1=kbd_getc();                  //Lee teclado y asigna a la variable b1.
        }
while(b1==0);                           //
output_high(PIN_B0);                    //Activa buffer.
delay_ms(200);                          //Espera 0,2 segundos.
output_low(PIN_B0);                     //Apaga buffer.
delay_ms(100);                          //Espera 0,1 segundos.

lcd_putc("\nTERCER NUMERO   ");         //Muestra "TERCER NUMERO".
```

Lee tercer número oprimido en el teclado y lo guarda en la variable c1.

```
Do{
 c1=kbd_getc();                         //Lee teclado y asigna a la variable c1.
        }
while(c1==0);                           //
output_high(PIN_B0);                    //Activa buffer.
delay_ms(200);                          //Espera 0,2 segundos.
output_low(PIN_B0);                     //Apaga buffer.
 delay_ms(100);                         //Espera 0,1 segundos.
```

Ahora realiza operación lógica de comparación y AND, entre las variables de memoria y las recién ingresadas, si el resultado es correcto permite ingresar a un nuevo bucle llamado rutina de selección.

```
if(a==a1&&b==b1&&c==c1)                 //Si a=4=a1 y b=3=b1 y c=7=c1.
        {                               //Realiza.
        delay_ms(300);                  //Espera 0,3 segundos.
        lcd_putc("\fCLAVE CORRECTA!!  ");//Muestra "CLAVE CORRECTA!!".
        lcd_putc("\nA TRABAJAR!!  ");   //Muestra "A TRABAJAR!!".
        delay_ms(2000);                 //Espera 2 segundos.
```

Muestra en la pantalla LCD los caracteres “CLAVE CORRECTA!!”, luego muestra, “LISTO!!!” y enciende los LEDs frontales.

```
        lcd_putc("\fLISTO");            //Muestra "LISTO".
        delay_ms(300);                  //Espera 0,3 segundos.
        lcd_putc("\fLISTO!");           //Muestra "LISTO!".
        delay_ms(300);                  //Espera 0,3 segundos.
        lcd_putc("\fLISTO!!");          //Muestra "LISTO!!".
        delay_ms(300);                  //Espera 0,3 segundos.
        lcd_putc("\fLISTO!!!");         //Muestra "LISTO!!!".
        output_high(PIN_B1);            //Enciende los LEDs frontales.
        delay_ms(1000);                 //Espera 1 segundos.
        output_low(PIN_B1);             //Apaga los LEDs

        while(1)
                {
                Rutina de selección de estación.
                }
 }
```

Si no es correcta la clave muestra “CLAVE INCORRECTA” y da tres “bips” y pide oprimir *, para volver a ingresar clave.

```
else
        {
        lcd_putc("\fCLAVE INCORRECTA");   //Muestra "CLAVE INCORRECTA".
        output_high(PIN_B0);              //Activa buffer.
        delay_ms(100);                    //Espera 0,1 segundos.
        output_low(PIN_B0);               //Apaga buffer.
        delay_ms(100);                    //Espera 0,1 segundos.
        output_high(PIN_B0);              //Activa buffer.
        delay_ms(100);                    //Espera 0,1 segundos.
        output_low(PIN_B0);               //Apaga buffer.
        delay_ms(100);                    //Espera 0,1 segundos.
        output_high(PIN_B0);              //Activa buffer.
        delay_ms(100);                    //Espera 0,1 segundos.
        output_low(PIN_B0);               //Apaga buffer.
        delay_ms(100);                    //Espera 0,1 segundos.
        lcd_putc("\nPresione *");         //Muestra "Presione *".
```

Lee carácter en el teclado y lo guarda en la variable r1.

```
        Do{
         r1=kbd_getc();                   //Lee teclado y asigna a la variable r1.
                    }
         while(r1!='*');                  //Si r1 es igual a * realiza y sale.
        output_high(PIN_B0);              //Activa buffer.
        delay_ms(200);                    //Espera 0,2 segundos.
        output_low(PIN_B0);               //Apaga buffer.
        delay_ms(100);                    //Espera 0,1 segundos.
         }
        }
```

Rutina de selección de estación.

Esta rutina permite al usuario escoger la estación a la que quiere enviar el prototipo y al igual que la anterior está dentro de un bucle infinito.

```
while(1)
      {
 lee:      lcd_putc("\f ELIJA ESTACION  ");   //Muestra "ELIJA ESTACION".
           lcd_putc("\n    A ENVIAR     ");   //Muestra "A ENVIAR".
           delay_ms(2000);                    //Espera 2 segundos.

            Do {
                    d=kbd_getc();             //Lee teclado y asigna a la variable d.
                    }
           while(d==0);
           output_high(PIN_B0);               //Activa buffer.
           delay_ms(200);                     //Espera 0,2 segundos.
           output_low(PIN_B0);                //Apaga buffer.
           delay_ms(100);                     //Espera 0,1 segundos.
```

Los cálculos matemáticos es mejor realizarlos entre números decimales en lo posible, ya que es más fácil la compresión y no permite malas interpretaciones, es por esta razón que se realiza la conversión de carácter a decimal.

```
e=(d-48);                          //Operación que convierte carácter a decimal.

printf(lcd_putc,"\fESTACION %d ",e);   //Muestra "ESTACION" y el número.
```

Muestra en la pantalla LCD “ESTACION” y el número elegido, seguido parpadean los LEDs el número de veces que corresponde a la estación.

```
for (k= 0; k < e; k++)             //Realiza incremento de k hasta k=e.
{
output_high(PIN_B1);               //Enciende los LEDs frontales.
delay_ms(300);                     //Espera 0,3 segundos.
output_low(PIN_B1);                //Apaga los LEDs frontales.
delay_ms(100);                     //Espera 0,1 segundos.
 }
```

Rutina de detección de obstáculos.

Entre las habilidades del prototipo está el detectar obstáculos en su recorrido y parar avance inmediatamente, dando alarma del hecho. Además al ocurrir este acontecimiento se visualiza en la pantalla LCD la distancia a la que se encuentra dicho obstáculo. Para la detección de objetos se usa el modulo sensor de ultrasonido SRF04.

El ultrasonido es un desplazamiento de partículas en el aire similar al del sonido, por tal razón lo que se realiza es generar un impulso de duración de 20 microsegundos, llamado tiempo de disparo. Luego se espera a recibir el reflejo de este impulso llamado eco y se cuenta el tiempo que se tarda.

```
        output_high(PIN_C4);       //Pulso para activar disparo.
        delay_us(20);              //Espera 20 microsegundos.
        output_low(PIN_C4);        //Apaga pulso de disparo.
 while(!input(PIN_C5))             //No recibe eco.
        {}                         //No hace nada.
        set_timer1(0);             //Activa temporizador 1.
while(input(PIN_C5))               //Recibe eco.
        {}                         //No hace nada.
         t1=get_timer1();          //Almacena tiempo en t1.
        d1=(t1*10)/(58.0);         //Realiza operación y guarda en d1.
```

Las variables d1 y t1, corresponden a distancia detectada y el tiempo que tarda en hacerlo. La fórmula para determinar esta distancia es:

$$d1 = \frac{t1*10}{58}$$

Seguido se pregunta la distancia es menor a 50, si los es para las demás actividades y muestra en la pantalla LCD, “PELIGRO!!! OBJETO A número de cm”, si la distancia es mayor sigue en las otras actividades.

```
if (d1<=50)
        {                               //Si d1 menor o igual a 50 cm.
        output_high(PIN_B0);            //Activa buffer.
        delay_ms(200);                  //Espera 0,2 segundos.
        output_low(PIN_B0);             //Apaga buffer.
        delay_ms(100);                  //Espera 0,1 segundos.
        output_low(PIN_B2);             //Apaga salida a puente H.
        output_low(PIN_B3);             //Apaga salida a puente H.
        output_low(PIN_B4);             //Apaga salida a puente H.
        output_low(PIN_B5);             //Apaga salida a puente H.

        printf(lcd_putc,"\fPELIGRO!!! \nOBJETO A cm:%Lu", d1);

        delay_ms(100);                  //Espera 0,1 segundos.
        }
```

Rutina de seguimiento de línea.

Basado en el programa de prueba **ROBOTESISMEJORADO.asm,** se ha hecho la rutina de control de seguimiento de línea, el diagrama de flujo es el mismo y obedece a un ciclo infinito.

Lo primero que se hace es leer las entradas análogas y pasar estos datos a valores digitales para ser examinados por el programa.

```
set_adc_channel(0);             //Lee canal 0.
delay_us(10);                   //Espera 10 microsegundos.
f=read_adc();                   //Almacena valor en f.

set_adc_channel(1);             //Lee canal 1.
delay_us(10);                   //Espera 10 microsegundos.
g=read_adc();                   //Almacena valor en g.

lcd_putc("\fSIGUIENDO CURSO");  //Muestra "SIGUIENDO CURSO".
printf(lcd_putc,"\nESTACION %d ",e);  //Muestra "ESTACION" y el número.
```

La rutina para decisión de control de los motoreductores, simplemente preguntar si los valores de f y g están dentro de un rango definido.

```
if(f < 100 && g < 100)                  //Atrás
        {                               //Si sensor derecho<150 y izquierdo<150.
        output_high(PIN_B1);            //Activa LEDs frontales.
        output_low(PIN_B2);             //Apaga salida a puente H.
        output_high(PIN_B3);            //Activa salida a puente H.
        output_high(PIN_B4);            //Activa salida a puente H.
        output_low(PIN_B5);             //Apaga salida a puente H.
        delay_ms(17);                   //Espera 0,017 segundos.
        output_low(PIN_B2);             //Apaga salida a puente H.
        output_low(PIN_B3);             //Apaga salida a puente H.
        output_low(PIN_B4);             //Apaga salida a puente H.
        output_low(PIN_B5);             //Apaga salida a puente H.
        delay_us(40);                   //Espera 40 microsegundos, tiempo de reposo.
         }
else
if(f > 100 && g < 100)                  //Izquierdo
        {                               //Si sensor derecho<150 y izquierdo<150.
        output_high(PIN_B1);            //Activa LEDs frontales.
        output_low(PIN_B2);             //Apaga salida a puente H.
        output_low(PIN_B3);             //Apaga salida a puente H.
        output_low(PIN_B4);             //Apaga salida a puente H.
        output_high(PIN_B5);            //Activa salida a puente H.
        delay_ms(17);                   //Espera 0,017 segundos.
        output_low(PIN_B2);             //Apaga salida a puente H.
        output_low(PIN_B3);             //Apaga salida a puente H.
        output_low(PIN_B4);             //Apaga salida a puente H.
        output_low(PIN_B5);             //Apaga salida a puente H.
        delay_us(40);                   //Espera 40 microsegundos, tiempo de reposo.
        }
else
if(f < 100 && g > 100)                  //Derecho
        {                               //Si sensor derecho<150 y izquierdo<150.
        output_high(PIN_B1);            //Activa LEDs frontales.
        output_high(PIN_B2);            //Activa salida a puente H.
        output_low(PIN_B3);             //Apaga salida a puente H.
        output_low(PIN_B4);             //Apaga salida a puente H.
        output_low(PIN_B5);             //Apaga salida a puente H.
        delay_ms(17);                   //Espera 0,017 segundos.
        output_low(PIN_B2);             //Apaga salida a puente H.
        output_low(PIN_B3);             //Apaga salida a puente H.
        output_low(PIN_B4);             //Apaga salida a puente H.
        output_low(PIN_B5);             //Apaga salida a puente H.
        delay_us(40);                   //Espera 40 microsegundos, tiempo de reposo.
        }
else
if(f > 100 && g > 100)                  //Adelante
        {                               //Si sensor derecho<150 y izquierdo<150.
        output_high(PIN_B1);            //Activa LEDs frontales.
        output_high(PIN_B2);            //Activa salida a puente H.
        output_low(PIN_B3);             //Apaga salida a puente H.
        output_low(PIN_B4);             //Apaga salida a puente H.
        output_high(PIN_B5);            //Activa salida a puente H.
        delay_ms(17);                   //Espera 0,017 segundos.
        output_low(PIN_B2);             //Apaga salida a puente H.
        output_low(PIN_B3);             //Apaga salida a puente H.
        output_low(PIN_B4);             //Apaga salida a puente H.
        output_low(PIN_B5);             //Apaga salida a puente H.
        delay_us(40);                   //Espera 40 microsegundos, tiempo de reposo.
        }
        }                               //Termina ciclo de seguidor.
```

3.3.3. Grabación en el microcontrolador

El primer programa desarrollado **ROBOTESISBASICO.asm**, es el indicado para realizar las pruebas del circuito electrónico y la plataforma mecánica. Teniendo el archivo **ROBOTESISBASICO.HEX**, desarrollado después de la compilación en el software Mplab, se debe abrir el software PICkit 2 dando doble clip en el icono del escritorio, como lo muestra la figura 87.

Figura 87. Icono de PICkit 2 en el escritorio.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

El archivo ejecutable o .hex es programado de forma directa al microcontrolador, donde se almacenará el programa en la memoria FLASH. Se usa el puerto USB del computador que se conecta al dispositivo grabador, al realizar esto se enciende el LED verde que indica que el grabador esta energizado.

Se inicia el programa Pickit2 y aparece la ventana de edición, figura 88. En ella se puede apreciar el menú de aplicación, está ubicado en la parte superior. El microcontrolador es identificado automáticamente por el programador, el USER ID y la palabra de configuración. En esta misma ventana se pueden ver el mensaje de estatus de la aplicación. En este caso se señala que el Pickit2 está conectado y que se reconoció al microcontrolador PIC16F887A.

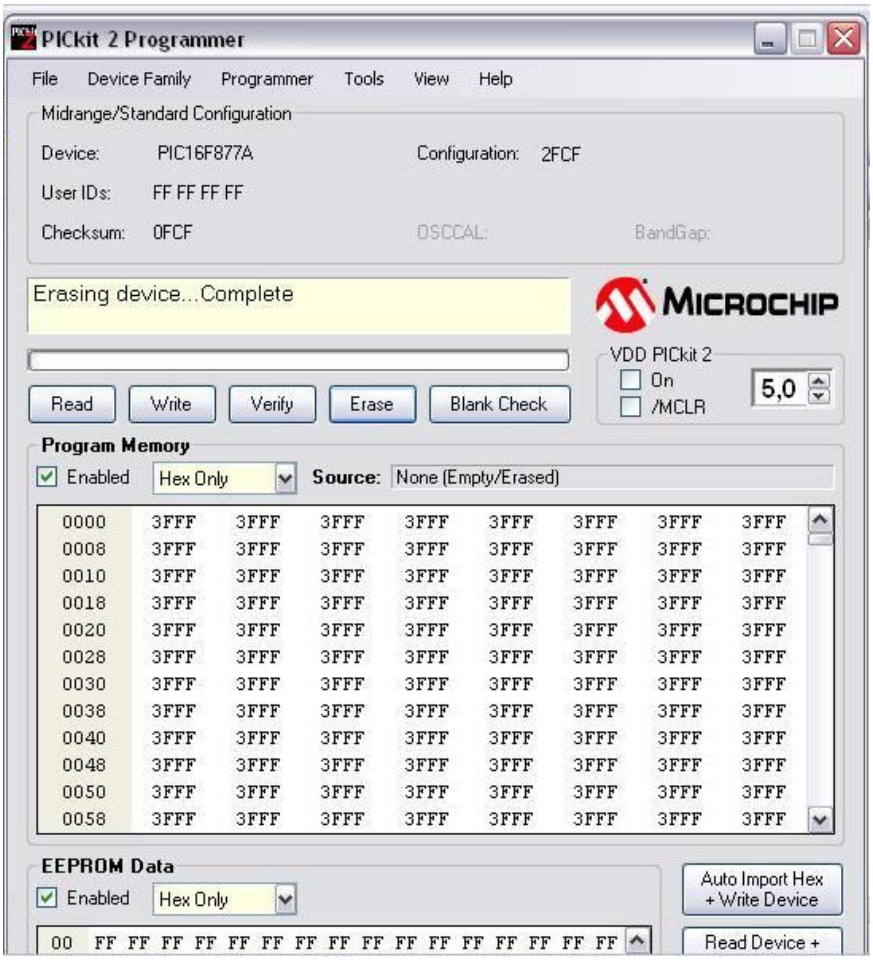


Figura 88. Ventana de edición Pickit2.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Además se muestra el voltaje de programación a utilizar, este voltaje podría servir como fuente de alimentación para el proyecto en el que este embebido el microcontrolador luego de la grabación; se muestra la memoria de programa del PIC, la memoria EEPROM del PIC. La ventana tiene los botones de programación, con ellos se puede:

- Leer el firmware actual que tiene almacenado el PIC en su memoria de programa, botón Read, al pulsar se despliega la codificación hexadecimal a la zona de program memory.

- Para bajar el firmware o mejor él .hex al PIC se usa el botón Write, este a su vez muestra el contenido desplegado en la zonas de: program memory, 'EEPROM data', MidRange/Standard configuration.

- Verificar el grabado correcto del firmware en el PIC, se usa el botón Verify.
- Para borrar la memoria de programa actual del PIC, se usa el botón Erase.
- Verificar que la memoria de programa actual del PIC este en blanco o borrada, se usa Blank Check.

El procedimiento para gravar el archivo .hex al microcontrolador es:

- Pulsar en el menú Programmer, se debe tener la opción Manual Device Select. Para que el Pickit2 identifique automáticamente el microcontrolador conectado al socket o protoboard.
- Pulsar en el menú Tools y verificar la conexión al microcontrolador, en Check Communication.
- Para que se reconozca de forma automática el microcontrolador que fue insertado en la PCB o en la protoboard, se despliega el mensaje PIC Device Found y el nombre del PIC reconocido, PIC16F887A.
- Se pulsa en el Menu File y luego en Import HEX, seleccionar el firmware a programar, archivo con extensión .hex, y el contenido del archivo se desplegara en la Program memory, EEPROM DATA; donde solo si la aplicación utiliza esta memoria.
- Se pulsa el boton Write, para bajar o grabar el archivo .hex en el microcontrolador. Los leds naranja Target y rojo Busy del Pickit2 se encienden mientras está en proceso la escritura. Si la programación fue exitosa se desplegara el mensaje Programming Successful.
- Por ultimo retirar el grabador del PCB o protoboard y probar si el microcontrolador se programó. Cerrar la aplicación Pickit 2.

3.3.4. Programa final

Este es el programa final. Resultado de las pruebas en el simulador Proteus, de las pruebas en protoboard y por último el comportamiento del prototipo robot en las pruebas de campo. El programa reúne los procedimientos de las rutinas antes explicadas y como característica de la programación, es susceptible a cambios para modificar su funcionalidad.

```
#include <CONTROLADOR.h>                    //Incluir CONTROLADOR .hex.
#include <LCD.C>                            //Incluir librería LCD.
#include <KBD.C>                            //Incluir librería Teclado de 3X4.

//Programa diseñado por Carlos Alberto Goyeneche Alfonso
//Estudiante de ingeniería electrónica.
//Universidad Nacional Abierta y a Distancia UNAD.
//Sogamoso 2016

void main()
{                                           //Programa principal.

  setup_adc_ports(AN0_AN1_AN3);             //Se usan los pines A0, A1 y A3 como entradas análogas.
  lcd_init();                               //Se inicializa Pantalla LCD.
  kbd_init();                               //Se inicializa Teclado matricial.

  char a='4', b='3', c='7';                 //Clave de acceso 437.
  char a1=0, b1=0, c1=0;                    //Variables que almacenan valores de la clave.
  char d;                                   //Variable  d=carácter estación.
  char m;                                   //Variable  m=carácter volver.
  char r1;                                  //Variable  r1=carácter clave incorrecta.
  int e;                                    //Variable  e=decimal estación.
  int f;                                    //Variable  f=sensor derecho.
  int g;                                    //Variable  g=sensor izquierdo.
  int h;                                    //Variable  h=contador de estaciones.
  int i;                                    //Variable  memoria de estación.
  int k;                                    //Variable del contador.
  int16 d1, t1;                             //Variables de ultrasonido d1=distancia t1=tiempo.

  setup_adc(ADC_CLOCK_DIV_4);               //Se ajusta el tiempo de conversión análoga.
  setup_timer_1(T1_INTERNAL|T1_DIV_BY_8);   //Se ajusta temporizador interno.

  lcd_putc("\f    INICIANDO    ");          //Muestra "INICIANDO".
  delay_ms(400);                            //Espera 0,4 segundos.
  lcd_putc("\n ---           ");           //Muestra 4 líneas.
  delay_ms(400);                            //Espera 0,4 segundos.
  lcd_putc("\n    ---        ");           //Muestra 4 líneas.
  delay_ms(400);                            //Espera 0,4 segundos.
  lcd_putc("\n       ---     ");           //Muestra 4 líneas.
  delay_ms(400);                            //Espera 0,4 segundos.
  lcd_putc("\n          --- ");            //Muestra 4 líneas.
  delay_ms(400);                            //Espera 0,4 segundos.
  lcd_putc("\fROBOT TRANSPORTE");           //Muestra "ROBOT TRANSPORTE".
  delay_ms(2000);                           //Espera 2 segundos.
  lcd_putc("\nCarlos Goyeneche");           //Muestra "Carlos Goyeneche".
  delay_ms(3000);                           //Espera 3 segundos.
  output_high(PIN_B1);                      //Enciende los LEDs frontales
  delay_ms(1000);                           //Espera 1 segundo
  output_low(PIN_B1);                       //Apaga los LEDs
  delay_ms(3000);                           //Espera 3 segundos.


while(1)
  {                                         //Ciclo de clave de acceso.
```

```
      a1=0;                                  //Se asigna 0 a variable a1.
      b1=0;                                  //Se asigna 0 a variable b1.
      c1=0;                                  //Se asigna 0 a variable c1.
      r1=0;                                  //Se asigna 0 a variable r1.
      h=0;                                   //Se asigna 0 a variable h.

      lcd_putc("\fCLAVE DE INGRESO");        //Muestra "CLAVE DE INGRESO".
      lcd_putc("\nPARA DIRECCIONAR");        //Muestra "PARA DIRECCIONAR".
      delay_ms(2000);                        //Espera 2 segundos.

      lcd_putc("\fINGRESE EL..");            //Muestra "INGRESE EL..".
      lcd_putc("\nPRIMER NUMERO   ");        //Muestra "PRIMER NUMERO".
    Do{
      a1=kbd_getc();                         //Lee teclado y asigna a la variable a1.
      }
      while(a1==0);                          //
      output_high(PIN_B0);                   //Activa buffer.
      delay_ms(200);                         //Espera 0,2 segundos.
      output_low(PIN_B0);                    //Apaga buffer.
      delay_ms(100);                         //Espera 0,1 segundos.

      lcd_putc("\nSEGUNDO NUMERO  ");        //Muestra "SEGUNDO NUMERO".
    Do{
      b1=kbd_getc();                         //Lee teclado y asigna a la variable b1.
      }
      while(b1==0);                          //
      output_high(PIN_B0);                   //Activa buffer.
      delay_ms(200);                         //Espera 0,2 segundos.
      output_low(PIN_B0);                    //Apaga buffer.
      delay_ms(100);                         //Espera 0,1 segundos.

    lcd_putc("\nTERCER NUMERO   ");          //Muestra "TERCER NUMERO".
    Do{
      c1=kbd_getc();                         //Lee teclado y asigna a la variable c1.
      }
      while(c1==0);                          //
      output_high(PIN_B0);                   //Activa buffer.
      delay_ms(200);                         //Espera 0,2 segundos.
      output_low(PIN_B0);                    //Apaga buffer.
      delay_ms(100);                         //Espera 0,1 segundos.

      if(a==a1&&b==b1&&c==c1)                //Si a=4=a1 y b=3=b1 y c=7=c1.
      {                                      //Muestra lo siguiente.
      delay_ms(300);                         //Espera 0,3 segundos.
      lcd_putc("\fCLAVE CORRECTA!!  ");      //Muestra "CLAVE CORRECTA!!".
      delay_ms(1000);                        //Espera 1 segundo.
      lcd_putc("\fA TRABAJAR!!  ");          //Muestra "A TRABAJAR!!".
      delay_ms(2000);                        //Espera 2 segundos.

  while(1)
      {
lee:  lcd_putc("\f ELIJA ESTACION  ");       //Muestra "ELIJA ESTACION".
      lcd_putc("\n    A ENVIAR     ");       //Muestra "A ENVIAR".
      delay_ms(1000);                        //Espera 1 segundo.

      Do {
      d=kbd_getc();                          //Lee teclado y asigna a la variable d.
        }
      while(d==0);
      output_high(PIN_B0);                   //Activa buffer.
      delay_ms(200);                         //Espera 0,2 segundos.
      output_low(PIN_B0);                    //Apaga buffer.
      delay_ms(100);                         //Espera 0,1 segundos.

      e=(d-48);                              //Operación que convierte carácter a decimal.

     if(d=='#')
     {                                       //Si d=#.
     goto dirigido;                          //Salta a dirigido.
```

```
        }
        if(d=='0')
        {                                                   //Si d=0.
        goto fuera;                                         //Salta a fuera.
        }

          if(d!='*')
          {                                                 //Si d=*.
            printf(lcd_putc,"\fESTACION %d ",e);            //Muestra "ESTACION" y el numero.
            for (k= 0; k < e; k++)                          //Realiza incremento de k hasta k=e.
              {
               output_high(PIN_B0);                         //Enciende los LEDs frontales.
               delay_ms(100);                               //Espera 0,3 segundos.
               output_low(PIN_B0);                          //Apaga los LEDs frontales.
               delay_ms(100);                               //Espera 0,1 segundos.
               }

 while(1)                                                   //Inicia rutina seguidor.
            {                           //
pregunt1:       output_high(PIN_C4);                        //Pulso para activar disparo.
               delay_us(20);                                //Espera 20 microsegundos.
               output_low(PIN_C4);                          //Apaga pulso de disparo.
            while(!input(PIN_C5))                           //No recibe eco.
               {}                                           //No hace nada.
               set_timer1(0);                               //Activa temporizador1.
            while(input(PIN_C5))                            //Recibe eco.
               {}                                           //No hace nada.
               t1=get_timer1();                             //Almacena tiempo en t1.
               d1=(t1*10)/(58.0);                           //Realiza operación y guarda en d1.
             if (d1<=50)
               {                                            //Si d1 menor o igual a 50 cm.
               output_high(PIN_B0);                         //Activa buffer.
               delay_ms(200);                               //Espera 0,2 segundos.
               output_low(PIN_B0);                          //Apaga buffer.
               delay_ms(100);                               //Espera 0,1 segundos.
               output_low(PIN_B2);                          //Apaga salida a puente H.
               output_low(PIN_B3);                          //Apaga salida a puente H.
               output_low(PIN_B4);                          //Apaga salida a puente H.
               output_low(PIN_B5);                          //Apaga salida a puente H.
               printf(lcd_putc,"\fPELIGRO!!! \nOBJETO A cm:%Lu", d1);
               delay_ms(100);                               //Espera 0,1 segundos.
            goto pregunt1;                                  //Salta a pregunt1.
               }

               set_adc_channel(0);                          //Lee canal 0.
               delay_us(10);                                //Espera 10 microsegundos.
               f=read_adc();                                //Almacena valor en f.

               set_adc_channel(1);                          //Lee canal 1.
               delay_us(10);                                //Espera 10 microsegundos.
               g=read_adc();                                //Almacena valor en g.

               lcd_putc("\fSIGUIENDO CURSO A ");            //Muestra "SIGUIENDO CURSO A".
             printf(lcd_putc,"\nESTACION %d ",h);           //Muestra "ESTACION" y el número.

              if(input(PIN_C2)==1)                          //Si la entrada RC2=1.
               {                                            //Corresponde a botón A del control.
               for (k= 0; k < e; k++)                       //Realiza incremento de k hasta k=e.
                {                                           //Para que active alarma.
                output_high(PIN_B0);                        //Enciende buffer.
                delay_ms(200);                              //Espera 0,2 segundos.
                output_low(PIN_B0);                         //Apaga buffer.
                delay_ms(100);                              //Espera 0,1 segundos.
                }
               }
              if(input(PIN_C3)==1)                          //Si la entrada RC3=1.
               {                                            //Corresponde a botón C del control.
               goto salida;                                 //Salta a salida.
               }
```

```
 if(f < 100 && g < 100)                  //Atrás
   {                                     //Si sensor derecho<150 y izquierdo<150.
   output_high(PIN_B1);                  //Activa LEDs frontales.
   output_low(PIN_B2);                   //Apaga.
   output_high(PIN_B3);                  //Activa.
   output_high(PIN_B4);                  //Activa.
   output_low(PIN_B5);                   //Apaga.
   delay_ms(50);                         //Espera 0,05 segundos.
   output_low(PIN_B2);                   //Apaga salida a puente H.
   output_low(PIN_B3);                   //Apaga salida a puente H.
   output_low(PIN_B4);                   //Apaga salida a puente H.
   output_low(PIN_B5);                   //Apaga salida a puente H.
   delay_us(40);                         //Espera 0,04 segundos, tiempo de reposo.
   }
 else
 if(f > 100 && g < 100)                  //Izquierdo
   {                                     //Si sensor derecho<150 y izquierdo<150.
   output_high(PIN_B1);                  //Activa LEDs frontales.
   output_low(PIN_B2);                   //Apaga.
   output_low(PIN_B3);                   //Apaga.
   output_low(PIN_B4);                   //Apaga.
   output_high(PIN_B5);                  //Activa.
   delay_ms(50);                         //Espera 0,05 segundos.
   output_low(PIN_B2);                   //Apaga salida a puente H.
   output_low(PIN_B3);                   //Apaga salida a puente H.
   output_low(PIN_B4);                   //Apaga salida a puente H.
   output_low(PIN_B5);                   //Apaga salida a puente H.
   delay_us(10);                         //Espera 0,01 segundos, tiempo de reposo.
   }
 else
 if(f < 100 && g > 100)                  //Derecho
   {                                     //Si sensor derecho<150 y izquierdo<150.
   output_high(PIN_B1);                  //Activa LEDs frontales.
   output_high(PIN_B2);                  //Activa.
   output_low(PIN_B3);                   //Apaga.
   output_low(PIN_B4);                   //Apaga.
   output_low(PIN_B5);                   //Apaga.
   delay_ms(50);                         //Espera 0,05 segundos.
   output_low(PIN_B2);                   //Apaga salida a puente H.
   output_low(PIN_B3);                   //Apaga salida a puente H.
   output_low(PIN_B4);                   //Apaga salida a puente H.
   output_low(PIN_B5);                   /Apaga salida a puente H.
   delay_us(10);                         //Espera 0,01 segundos, tiempo de reposo.
   }
 else
 if(f > 100 && g > 100)                  //Adelante
   {                                     //Si sensor derecho<150 y izquierdo<150.
   output_high(PIN_B1);                  //Activa LEDs frontales.
   output_high(PIN_B2);                  //Activa.
   output_low(PIN_B3);                   //Apaga.
   output_low(PIN_B4);                   //Apaga.
   output_high(PIN_B5);                  //Activa.
   delay_ms(50);                         //Espera 0,05 segundos.
   output_low(PIN_B2);                   //Apaga salida a puente H.
   output_low(PIN_B3);                   //Apaga salida a puente H.
   output_low(PIN_B4);                   //Apaga salida a puente H.
   output_low(PIN_B5);                   //Apaga salida a puente H.
   delay_us(40);                         //Espera 0,04 segundos, tiempo de reposo.
   }

 if(input(PIN_A2)==1)                    //Pregunta si RA2=1.
   {
    output_high(PIN_B0);                 //Activa buffer.
    output_high(PIN_B1);                 //Activa LEDs frontales.
    output_high(PIN_B2);                 //Activa.
    output_low(PIN_B3);                  //Apaga.
    output_low(PIN_B4);                  //Apaga.
    output_high(PIN_B5);                 //Activa.
    delay_ms(50);
```

```
                h=h+1;                                  //Incrementa h.
                if(h==e)                                //Pregunta si h=e.
                  {
                  output_low(PIN_B1);                   //apaga LEDs color
                  output_low(PIN_B2);                   //Apaga.
                  output_low(PIN_B3);                   //Apaga.
                  output_low(PIN_B4);                   //Apaga.
                  output_low(PIN_B5);                   //Apaga.
                  for (k= 0; k < 7; k++)                //Realiza incremento de k hasta k=e.
                  {
                  output_high(PIN_B0);                  //Activa buffer.
                  delay_ms(100);                        //Espera 0,1 segundos.
                  output_low(PIN_B0);                   //Apaga buffer.
                  delay_ms(100);                        //Espera 0,1 segundos.
                  }
                  goto almacen;                         //Salta a almacén.
                  }
                delay_ms(150);                          //Espera 0,15 segundos.
                output_low(PIN_B0);                     //Apaga buffer.
                output_low(PIN_B2);                     //Apaga salida a puente H.
                output_low(PIN_B3);                     //Apaga salida a puente H.
                output_low(PIN_B4);                     //Apaga salida a puente H.
                output_low(PIN_B5);                     //Apaga salida a puente H.
                delay_us(30);                           //Espera 0,03 segundos, tiempo de reposo.
                }
                }                                       //Termina ciclo de seguidor.
               }

fuera:          printf(lcd_putc,"\fOPCION NO \nHABILITADA!!");
               delay_ms(2000);                          //Espera 2 segundos.
               }

almacen:              lcd_putc("\fPARA VOLVER  \nPresione *");
                    delay_ms(2000);                     //Espera 2 segundos.
                    i=10-e;                             //Operación para contar.

 pregunte:            Do {
                    m=kbd_getc();                       //Lee teclado y asigna a la variable m.
                    }
                   while(m==0);
                   output_high(PIN_B0);                 //Activa buffer.
                   delay_ms(200);                       //Espera 0,2 segundos.
                   output_low(PIN_B0);                  //Apaga buffer.
                   delay_ms(100);                       //Espera 0,1 segundos.
                    if(m!='*')
                      {
                   goto pregunte;                       //Salta a pregunte.
                      }

              while(1)
              {                                         //Inicia rutina seguidor.
pregunt2:         output_high(PIN_C4);                  //Pulso para activar disparo.
               delay_us(20);                            //Espera 20 microsegundos.
               output_low(PIN_C4);                      //Apaga pulso de disparo.
             while(!input(PIN_C5))                      //No recibe eco.
               {}                                       //No hace nada.
               set_timer1(0);                           //Activa temporizador1.
             while(input(PIN_C5))                       //Recibe eco.
               {}                                       //No hace nada.
               t1=get_timer1();                         //Almacena tiempo en t1.
               d1=(t1*10)/(58.0);                       //Realiza operación y guarda en d1.
            if (d1<=50)
               {                                        //Si d1 menor o igual a 50 cm.
               output_high(PIN_B0);                     //Activa buffer.
               delay_ms(200);                           //Espera 0,2 segundos.
               output_low(PIN_B0);                      //Apaga buffer.
               delay_ms(100);                           //Espera 0,1 segundos.
               output_low(PIN_B2);                      //Apaga salida a puente H.
```

```
  output_low(PIN_B3);                          //Apaga salida a puente H.
  output_low(PIN_B4);                          //Apaga salida a puente H.
  output_low(PIN_B5);                          //Apaga salida a puente H.
  printf(lcd_putc,"\fPELIGRO!!! \nOBJETO A cm:%Lu", d1);
  delay_ms(100);                               //Espera 0,1 segundos.
   goto pregunt2;                              //Salta a pregunt1.
  }

  set_adc_channel(0);                          //Lee canal 0.
  delay_us(10);                                //Espera 10 microsegundos.
  f=read_adc();                                //Almacena valor en f.

  set_adc_channel(1);                          //Lee canal 1.
  delay_us(10);                                //Espera 10 microsegundos.
  g=read_adc();                                //Almacena valor en g.

  lcd_putc("\fSIGUIENDO CURSO A");             //Muestra "SIGUIENDO CURSO A".
  printf(lcd_putc,"\nALMACEN %d ",i);          //Muestra "ALMACEN" y el número.

 if(input(PIN_C2)==1)                          //Si la entrada RC2=1.
  {                                            //Corresponde a boton A.
  for (k= 0; k < e; k++)                       //Realiza incremento de k hasta k=e.
   {                                           //Para que active alarma.
   output_high(PIN_B0);                        //Enciende buffer.
   delay_ms(200);                              //Espera 0,2 segundos.
   output_low(PIN_B0);                         //Apaga buffer.
   delay_ms(100);                              //Espera 0,1 segundos.
   }
  }
 if(input(PIN_C3)==1)                          //Si la entrada RC3=1.
  {                                            //Corresponde a boton C del control.
  goto salida;                                 //Salta a salida.
  }
 if(f < 100 && g < 100)                        //Atras
  {                                            //Si sensor derecho<150 y izquierdo<150.
  output_high(PIN_B1);                         //Activa LEDs frontales.
  output_low(PIN_B2);                          //Apaga.
  output_high(PIN_B3);                         //Activa.
  output_high(PIN_B4);                         //Activa.
  output_low(PIN_B5);                          //Apaga.
  delay_ms(50);                                //Espera 0,05 segundos.
  output_low(PIN_B2);                          //Apaga salida a puente H.
  output_low(PIN_B3);                          //Apaga salida a puente H.
  output_low(PIN_B4);                          //Apaga salida a puente H.
  output_low(PIN_B5);                          //Apaga salida a puente H.
  delay_us(40);                                //Espera 0,04 segundos, tiempo de reposo.
  }
else
 if(f > 100 && g < 100)                        //Izquierdo
  {                                            //Si sensor derecho<150 y izquierdo<150.
  output_high(PIN_B1);                         //Activa LEDs frontales.
  output_low(PIN_B2);                          //Apaga.
  output_low(PIN_B3);                          //Apaga.
  output_low(PIN_B4);                          //Apaga.
  output_high(PIN_B5);                         //Activa.
  delay_ms(50);                                //Espera 0,05 segundos.
  output_low(PIN_B2);                          //Apaga salida a puente H.
  output_low(PIN_B3);                          //Apaga salida a puente H.
  output_low(PIN_B4);                          //Apaga salida a puente H.
  output_low(PIN_B5);                          //Apaga salida a puente H.
  delay_us(10);                                //Espera 0,01 segundos, tiempo de reposo.
  }
else
 if(f < 100 && g > 100)                        //Derecho
  {                                            //Si sensor derecho<150 y izquierdo<150.
  output_high(PIN_B1);                         //Activa LEDs frontales.
  output_high(PIN_B2);                         //Activa.
  output_low(PIN_B3);                          //Apaga.
  output_low(PIN_B4);                          //Apaga.
```

```
                output_low(PIN_B5);                 //Apaga.
                delay_ms(50);                       //Espera 0,05 segundos.
                output_low(PIN_B2);                 //Apaga salida a puente H.
                output_low(PIN_B3);                 //Apaga salida a puente H.
                output_low(PIN_B4);                 //Apaga salida a puente H.
                output_low(PIN_B5);                 //Apaga salida a puente H.
                delay_us(10);                       //Espera 0,01 segundos, tiempo de reposo.
                }
              else
               if(f > 100 && g > 100)               //Adelante
                {                                   //Si sensor derecho<150 y izquierdo<150.
                output_high(PIN_B1);                //Activa LEDs frontales.
                output_high(PIN_B2);                //Activa.
                output_low(PIN_B3);                 //Apaga.
                output_low(PIN_B4);                 //Apaga.
                output_high(PIN_B5);                //Activa.
                delay_ms(50);                       //Espera 0,05 segundos.
                output_low(PIN_B2);                 //Apaga salida a puente H.
                output_low(PIN_B3);                 //Apaga salida a puente H.
                output_low(PIN_B4);                 //Apaga salida a puente H.
                output_low(PIN_B5);                 //Apaga salida a puente H.
                delay_us(40);                       //Espera 0,04 segundos, tiempo de reposo.
                }

               if(input(PIN_A2)==1)                 //Pregunta si RA2=1.
                {
                 output_high(PIN_B0);               //Activa buffer.
                 output_high(PIN_B1);               //Activa LEDs frontales.
                 output_high(PIN_B2);               //Activa.
                 output_low(PIN_B3);                //Apaga.
                 output_low(PIN_B4);                //Apaga.
                 output_high(PIN_B5);               //Activa.
                 delay_ms(50);

                 i=i-1;                             //Decrementa i.
                 if(i==0)                           //Pregunta si i=0.
                   {
                   output_low(PIN_B1);              //Apaga LEDs color
                   output_low(PIN_B2);              //Apaga.
                   output_low(PIN_B3);              //Apaga.
                   output_low(PIN_B4);              //Apaga.
                   output_low(PIN_B5);              //Apaga.
                   for (k= 0; k < 7; k++)           //Realiza incremento de k hasta k=7.
                   {
                   output_high(PIN_B0);             //Activa buffer.
                   delay_ms(100);                   //Espera 0,3 segundos.
                   output_low(PIN_B0);              //Apaga buffer.
                   delay_ms(100);                   //Espera 0,1 segundos.
                   }
                   goto salida;                     //Salta a salida.
                   }
                 delay_ms(150);                     //Espera 0,15 segundos.
                 output_low(PIN_B0);                //Apaga buffer.
                 output_low(PIN_B2);                //Apaga salida a puente H.
                 output_low(PIN_B3);                //Apaga salida a puente H.
                 output_low(PIN_B4);                //Apaga salida a puente H.
                 output_low(PIN_B5);                //Apaga salida a puente H.
                 delay_us(30);                      //Espera 0,03 segundos, tiempo de reposo.
                 }
                }

salida:    lcd_putc("\fMISION CUMPLIDA \nOTRA TAREA...");
           delay_ms(3000);                          //Espera 3 segundos.
           }
          else
           {
           lcd_putc("\fCLAVE INCORRECTA");          //Muestra "CLAVE INCORRECTA".
           for (k= 0; k < 5; k++)                   //Realiza incremento de k hasta k=5.
                 {
```

```
                output_high(PIN_B0);                    //Activa buffer.
                delay_ms(100);                          //Espera 0,3 segundos.
                output_low(PIN_B0);                     //Apaga buffer.
                delay_ms(100);                          //Espera 0,1 segundos.
                }
          lcd_putc("\nPresione *");                     //Muestra "Presione *".
        Do{
          r1=kbd_getc();                                //Lee teclado y asigna a la variable r1.
          }
          while(r1!='*');                               //Si r1 es igual a * realiza y sale.
          output_high(PIN_B0);                          //Activa buffer.
          delay_ms(200);                                //Espera 0,2 segundos.
          output_low(PIN_B0);                           //Apaga buffer.
          delay_ms(100);                                //Espera 0,1 segundos.
          }
}

dirijido:
          while(1)
          {
          lcd_putc("\fCONTROL MANUAL  ");               //Muestra "CONTROL MANUAL".
          delay_ms(10);
          if(input(PIN_C0)==1)                          //Si la entrada RC3=1.
            {                                           //Corresponde a botón D.
            output_high(PIN_B2);                        //Activa salida a puente H.
            delay_ms(20);                               //Espera 0,02 segundos.
            output_low(PIN_B2);                         //Apaga salida a puente H.
            delay_us(10);                               //Espera 0,01 segundos, tiempo de reposo.
            }
          if(input(PIN_C1)==1)                          //Si la entrada RC3=1.
            {                                           //Corresponde a botón B.
            output_high(PIN_B5);                        //Activa salida a puente H.
            delay_ms(20);                               //Espera 0,02 segundos.
            output_low(PIN_B5);                         //Apaga salida a puente H.
            delay_us(10);                               //Espera 0,01 segundos, tiempo de reposo.
            }
          if(input(PIN_C2)==1)                          //Si la entrada RC3=1.
            {                                           //Corresponde a botón A.
            output_high(PIN_B2);                        //Activa salida a puente H.
            output_high(PIN_B5);                        //Activa salida a puente H.
            delay_ms(20);                               //Espera 0,02 segundos.
            output_low(PIN_B2);                         //Apaga salida a puente H.
            output_low(PIN_B5);                         //Apaga salida a puente H.
            delay_us(10);                               //Espera 0,01 segundos, tiempo de reposo.
            }
          if(input(PIN_C3)==1)                          //Si la entrada RC3=1.
            {                                           //Corresponde a botón C.
            output_high(PIN_B3);                        //Activa salida a puente H.
            output_high(PIN_B4);                        //Activa salida a puente H.
            delay_ms(20);                               //Espera 0,02 segundos.
            output_low(PIN_B3);                         //Apaga salida a puente H.
            output_low(PIN_B4);                         //Apaga salida a puente H.
            delay_us(10);                               //Espera 0,01 segundos, tiempo de reposo.
            }
      if((input(PIN_C2)==1)&&(input(PIN_C3)==1))//Si la entrada RC3=1.(a==a1&&b==b1&&c==c1)
            {                                           //Corresponde a botón C.
            goto salida;                                //Salta a salida.
            }
          }
}                                                       //Fin.
```

4. USO Y MANTENIMIENTO

4.1. GUIA DE USO DEL PROTOTIPO

Antes debemos recordar, que el prototipo es solo para fines de experimentación. Este puede cargar sobre sí mismo, un peso de aproximadamente 3 kilogramos; su velocidad es de aproximadamente 0,25 metros por segundo. Y aunque el prototipo es muy fácil de maniobrar, se puede llegar a cometer algunos errores, si se desconocen los pasos de uso.

Nota: Se deben cargar las baterías internas del prototipo antes de usar, aproximadamente 4 horas, así garantiza el funcionamiento en las prácticas. El voltaje de carga es de 12 voltios DC y una corriente de carga máxima de 1500 miliamperios por hora.

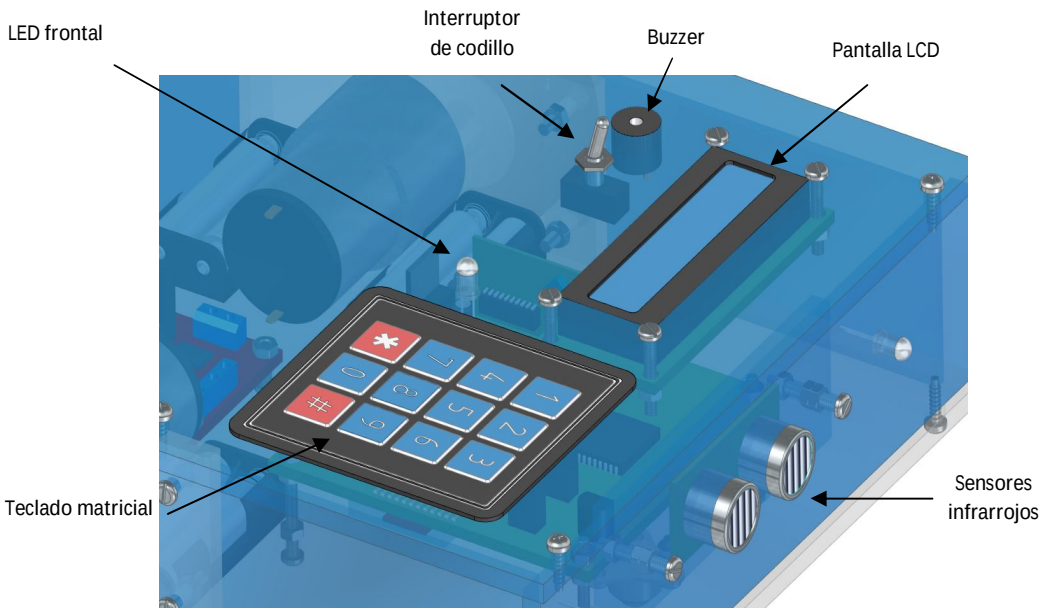


Figura 89. Panel frontal del prototipo.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

La figura 89 muestra un dibujo del panel frontal, formado por un interruptor de codillo, una pantalla LCD, el teclado matricial y el buzzer. Gracias a estos elementos el usuario u operador puede interactuar con el prototipo y darle instrucciones.

La operación y funcionamiento del prototipo se describe paso a paso a continuación:

1. Para encender el prototipo se debe mover el interruptor de codillo atrás. Se ilumina la pantalla LCD y aparece la palabra **INICIANDO**. En segunda línea se desplazan tres rayitas de izquierda a derecha.

2. Luego sale la presentación **ROBOT TRANSPORTE**. Seguido de **Carlos Goyeneche** (nombre de su servidor). Se encienden tres veces los Leds frontales bicolor y espera tres segundos.

3. La pantalla presenta el mensaje **CLAVE DE INGRESO**. En la segunda línea **PARA DIRECCIONAR**.

4. La pantalla muestra **INGRESE EL… PRIMER NUMERO**. Se debe oprimir el primer número de la clave de tres dígitos. Suena un bip al oprimir el teclado y enciende el Led verde frontal.

5. La pantalla muestra **INGRESE EL… SEGUNDO NUMERO**. Se debe oprimir el segundo número de la clave de tres dígitos. Suena un bip al oprimir el teclado y enciende el Led verde frontal.

6. La pantalla muestra **INGRESE EL… TERCER NUMERO**. Se debe oprimir el tercer número de la clave de tres dígitos. Suena un bip al oprimir el teclado y enciende el Led verde frontal.

7. La pantalla muestra **CLAVE CORRECTA!!** Si la misma corresponde. Luego muestra A **TRABAJAR!!** En este momento es que el usuario carga sobre el prototipo robot, los materiales a enviar. En este momento el prototipo robot está en reposo.

8. La pantalla muestra **ELIJA ESTACION**, en la segunda línea muestra **A ENVIAR**. Se escoge la estación oprimiendo el número en el teclado. Al realizarlo suenan los bips correspondientes al número seleccionado. Ejemplo 4, suenan cuatro bips y el led verde frontal se enciende al tiempo con el bip.

9. Cuando inicia recorrido, muestra **SIGUIENDO CURSO**. En la segunda línea **ESTACION** y el numero escogido. El prototipo realiza su desplazamiento, siguiendo la línea y cuenta las marcas que le indican frente a cual estación se encuentra, al llegar a la estación direccionada se detiene.

10. Al detenerse muestra la pantalla **PARA VOLVER** y en la segunda línea **Presione ***.

11. Si durante su recorrido encuentra algún objeto al frente a una distancia de aproximadamente 50 centímetros, el prototipo se detiene, y la pantalla

muestra **PELIGRO!!**, en la segunda línea **OBJETO A cm:** (distancia al objeto), además suena el bip repetidas veces y enciende el led verde frontal, dando alarma al operador. El prototipo robot no avanzará hasta que el obstáculo sea retirado.

12. Una vez el prototipo ha llegado a la estación, este se detiene. El operador de la estación retira los elementos enviados y lo debe regresar al almacén. Al presionar en el teclado *, el prototipo inicia su marcha hacia el almacén. La pantalla muestra **SIGUIENDO CURSO**. En la segunda línea **ESTACION** y el número de la estación frente a la que pasa.

13. Al momento de llegar al almacén, muestra la pantalla **MISION CUMPLIDA**, en la segunda línea **OTRA TAREA…**. Después de realizar un recorrido el prototipo queda en reposo y si se desea dar un nuevo recorrido es necesario volver a introducir la clave, se repiten los pasos 4, 5, 6, 7, 8, 9 y 10.

14. Si la clave no corresponde, muestra **CLAVE INCORRECTA**. El buzzer realiza tres bips y enciende el led verde frontal. En la segunda línea **Presione ***. Se debe oprimir la tecla * y se repiten los pasos 3, 4, 5, y 6.

Manejo del prototipo robot con el control remoto

Para que el prototipo responda teledirigidamente, se cuenta con un pequeño control remoto, figura 90. Este dispositivo le permite al operador detener el prototipo, da alarma y reiniciar el programa, para el modo automático. En modo manual le permite controlar sus movimientos atrás, adelante, derecha e izquierda.

Figura 90. Fotografía del control remoto.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

El control remoto tiene cuatro pequeños botones, A, B, C y D. Estos realizan lo siguiente:

Modo automático.		**Modo manual.**	
A	Detiene y da alarma.	**A**	Adelante.
B	No asignado.	**B**	Derecha.
C	Termina trabajo.	**C**	Atrás.
D	No asignado.	**D**	Izquierda.
		A y **C**	Sale modo manual.

En modo automático, el operador puede parar el prototipo a distancia y dar alarma, si advierte un peligro, o simplemente para dar un aviso, con el botón **A**. También puede detener la programación y simplemente bloquear el prototipo, con el botón **C**.

En modo manual, el operador puede controlar el prototipo a criterio. Los sensores quedan deshabilitados, al igual que el teclado y el buzzer. Para salir de modo manual, se deben oprimir los botones en el control, **A** y **C** al tiempo.

Nota: Si se presenta un reinicio del programa durante un recorrido a una estación, el prototipo no memoriza dicha asignación y empieza con sus registros en cero. El operador debe mover de manera manual el prototipo hasta el almacén y empezar de nuevo. Por esta razón, no se recomienda pulsar la tecla **C**, a menos que se trate de una emergencia. El botón **A** en modo automático, detiene el prototipo y dura en reposo solo el tiempo, que el operador tenga oprimida la tecla, luego avanza de manera normal.

Advertencia: el prototipo solo realiza una asignación de tarea y una vez finalizada dicha tarea, se debe ingresar la clave de acceso. Si no se tiene dicha clave, no es posible operar el prototipo, ya sea en modo automático o en modo manual. **“Sin la clave no trabaja”**. La clave está asignada en el programa y no es posible cambiarla de manera externa. La clave del prototipo robot es: **4** cuatro, **3** tres y **7** siete.

4.2. MANTENIMIENTO DEL PROTOTIPO

Mantenimiento, se puede definir como el conjunto de actividades que se realizan a un sistema, equipo o componente para asegurar que continúe desempeñando las funciones deseadas dentro de los parámetros óptimos. Se debe buscar que el prototipo cumpla con el objetivo para el cual fue diseñado; es por esta razón que se cuenta con algunas consideraciones básicas de uso.

Limpieza: Es una de las actividades de conservación de equipos más sencilla, que es simplemente retirar suciedad, polvo, residuos y todo tipo de materia extraña que se adhiere al prototipo.

Inspección: Sirve para averiguar el estado real del equipo, consiste en examinar de manera visual todas las partes que componen el prototipo, verificando que estén en buen estado y que funcionen correctamente.

Ajuste: Conocido como calibración, forma parte de los trabajos de conservación del prototipo. Consiste en reestablecer las condiciones de funcionamiento iniciales por ejemplo: apretar tornillos flojos, verificar cables. De esta forma se evitan fallas de funcionamiento.

Una gran característica de los equipos electrónicos, es su confiabilidad, gracias a su alta disposición de servicio y tiempos cortos de mantenimiento. Se debe buscar que las intervenciones al prototipo sean realizadas por personas especializadas y de ser necesario reemplazar componentes, estos últimos sean de alta calidad.

4.2.1. Limpieza

Se debe realizar antes de usar el prototipo, tiene un tiempo estimado de cinco minutos. Para esta actividad se deben usar las siguientes herramientas:

- Brocha pequeña de cerdas finas.
- Trapo que no suelte motas, por ejemplo franela.

Procedimiento:

- Apagar el prototipo robot.
- Retirar suavemente el polvo usando la brocha en las zonas con hendiduras.
- Luego con el trapo limpiar las áreas planas frotando suavemente.
- Importante no usar disolventes ya que estos pueden dañar los componentes de acrílico.
- Revisar la apariencia física de los componentes del prototipo robot.
- Por ultimo energizar para verificar funcionalidad.

4.2.2. Inspección

La inspección debe ser hecha por personal con conocimientos de electrónica y en ella se requiere el uso de herramientas de mano como: destornilladores y multímetro. Vea la **figura 100** para más detalles.

Procedimiento:

- Realizar primero etapa de limpieza.
- Apagar el prototipo robot.
- Retirar los seis tornillos de la tapa superior.
- Verificar si las baterías recargables están sueltas y/o sulfatadas.
- Medir voltaje de cada una de las baterías.
- Verificar conexiones de los cables.
- Verificar apriete de los tornillos.
- Encender prototipo y ver que la pantalla LCD, el teclado matricial y los diodos LED funcionan correctamente.
- Apagar e instalar tapa superior, cuidando de no dañar los cables.

- Encender y probar funcionamiento.
- Documentar inspección.

4.2.3. Ajuste

El ajuste del prototipo robot debe ser hecho por personal con conocimientos en electrónica, con experiencia en microcontroladores y programación.

Procedimiento:

- Realizar primero etapa de limpieza externa del prototipo antes de intervenir.
- Seguido realizar la etapa de inspección visual externa del prototipo.
- Apagar el prototipo, retirar las baterías, para evitar daños eléctricos por un mal contacto y generar corto circuito.
- Retirar la tapa, soltando los seis tornillos.
- Realizar aseo interno del prototipo usando soplador manual.
- Soltar los cables que llegan a la tarjeta PCB, marcándolos para no equivocar su posición.
- Soltar la tarjeta PCB de los cuatro tornillos y limpiar con brocha los componentes. Por el lado de las soldaduras limpiar con thinner y cepillo plástico para remover residuos de pomada o sulfato generado por humedad. Si es necesario retocar soldaduras sueltas (seguir procedimiento de soldadura, descrito en este documento).
- Aplicar limpiador de contacto a los terminales de cada cable y a los espadines.
- Instalar la tarjeta PCB y apretar bien las tuercas.
- Conectar los cables, fijar los conectores con silicona, para evitar que se suelten.

- Medir continuidad en el interruptor de codillo, para descartar malos contactos.

- Medir continuidad entre las diferentes tarjetas, verificando cada cable por separado.

- Luego medir todas las baterías, reemplazar las que estén deficientes por nuevas, instalar las baterías en los porta pilas.

- Verificar los amarres plásticos de los cables, y retirar los que estén defectuosos.

- Encender y verificar voltajes. Probar que la pantalla muestre los mensajes y el teclado funcione correctamente.

- Apagar y tapar, ajustando los tornillos de la tapa, teniendo en cuenta no dañar los cables.

- Verificar apriete de todos los tornillos para evitar desajuste de la plataforma, pero no aplicar mucha fuerza, ya que se trata de acrílico muy fácil de romper.

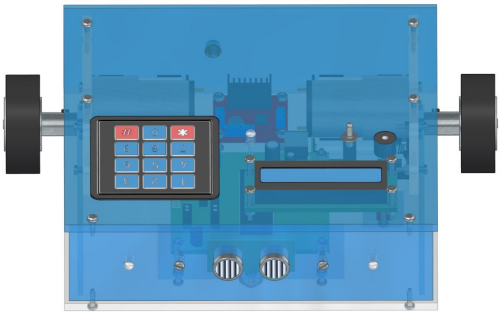
- Verificar encendido y funcionamiento del prototipo, ensayarlo en una pista. Probar todas sus funciones.

De ser necesario modificar el programa, a causa de una mejora, se debe realizar primero una copia de seguridad al programa original; ya que de no resultar exitosas las modificaciones, se puede cargar el programa y no tener inconvenientes. En este caso es necesario tener el archivo .C, del programa, el software CCS y el grabador de microcontroladores.

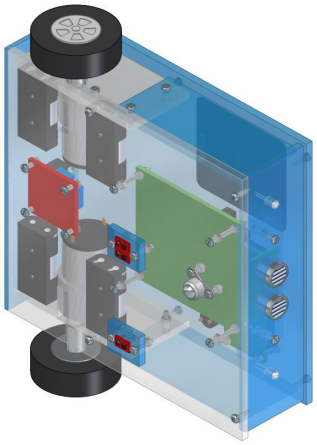
Nota: el mantenimiento del prototipo garantiza su buen desempeño y alarga su vida útil. Pero intervenir muy seguido, genera desgaste en algunas piezas, por ejemplo: los orificios de los tornillos. Se recomienda una limpieza diaria, una inspección mensual y un ajuste anual.

Advertencia: se deben tener en cuenta todas las medidas de seguridad eléctrica, al momento de intervenir el prototipo robot, ya que se pueden ocasionar daños al equipo y lesiones al personal de mantenimiento.

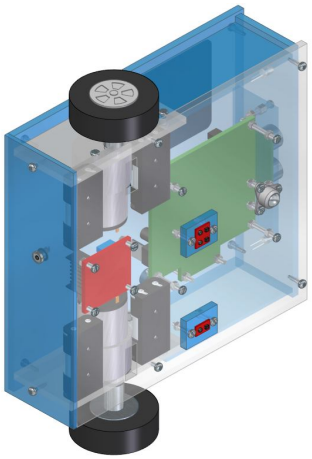
La figura 91, muestra los dibujos ilustrando el procedimiento de inspección del prototipo robot.



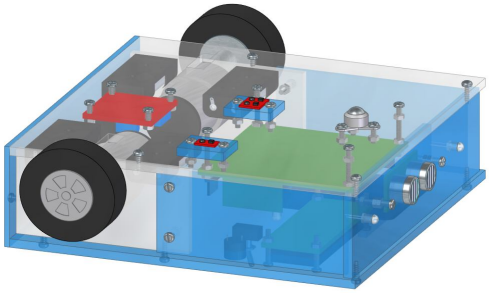
1. Verificar por arriba.



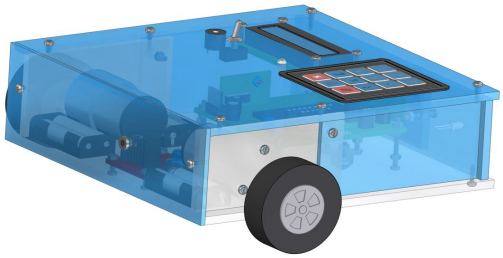
2. Verificar por la derecha



3. Verificar por la izquierda.



4. Verificar por debajo.



5. Verificar por atrás.

Figura 91. Dibujos procedimiento de inspección.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

5. ANÁLISIS ECONÓMICO

En la construcción del prototipo robot, se plantea la utilización de componentes y materiales de bajo costo y fáciles de adquirir. Se hace uso de herramientas de mano básicas, de manera se pueda armar una copia siguiendo los datos consignados en este documento.

5.1. EQUIPOS Y COMPONENTES

- Computador portátil, usado para realizar la simulación, los dibujos y el trabajo escrito. En la simulación es necesario la instalación de Proteus Isis, Proteus Ares, Pickit2, CCS Compiler, MPlab, Solid Edge y el paquete de office.
- Grabador de microcontroladores Pickit2, el cual incluye el programa de instalación.
- Componentes electrónicos para el prototipo, ver **tabla 4**.
- Materiales para la construcción del prototipo robot, ver **tabla 5**.

5.2. PRESUPUESTO

El costo de la investigación, diseño y construcción del proyecto, así como las herramientas y materiales adicionales necesarios, fueron soportados por el estudiante Carlos Alberto Goyeneche Alfonso.

Los componentes electrónicos en su mayoría fueron suministrados por la tienda de electrónica Mercatronics de Sogamoso.

Costo de componentes electrónicos.

Ítem	Componentes	Cantidad	Costo unidad	Costo total ítem
1	Microcontrolador PIC16F877A	2 un	$12.000	$24.000
2	Cristal 4MHz	2 un	$ 3.500	$7.000
3	Condensadores de 22 picos	4 un	$150	$600
4	Condensadores de 100 nanos	4 un	$150	$600
5	Condensadores de 100 micros	4 un	$150	$600
6	Resistencia 10k ohmios	30 un	$50	$1.500
7	Resistencia 1k ohmios	30 un	$50	$1.500
8	Resistencia de 330 ohmios	30 un	$50	$1.500
9	Resistencia 5,6k ohmios	30 un	$50	$1.500
10	Pulsador de board	2 un	$1.000	$2.000
11	Diodo LED bicolor	4 un	$1.500	$6.000
12	Diodo LED verde	4 un	$500	$2.000
13	Interruptor de codillo	2 un	$4.000	$8.000
14	Pantalla LCD	2 un	$20.000	$40.000
15	Teclado matricial	2 un	$10.000	$20.000
16	Porta pilas AAA	4 un	$2.000	$8.000
17	Sensor HC-SR04	2 un	$12.000	$24.000
18	Sensor QTR-1A	4 un	$8.000	$32.000
19	Cables con terminal	50 un	$300	$15.000
20	Reglilla de espadines	4 un	$3.000	$12.000
21	Regulador 7805	2 un	$5.000	$10.000
22	Modulo RF M40 RFC	2 un	$35.000	$70.000
23	Buzzer	2 un	$3.000	$6.000
24	Baquelita 10X10	2 un	$5.000	$10.000
25	Modulo Puente H	1 un	$38.000	$38.000
26	Moto reductores	2 un	$45.000	$90.000
27	Trimmer 5 K	2 un	$3.000	$6.000
28	Plug pasamuros	2 un	$2.000	$4.000
29	Transistor 2N3904	2 un	$1.500	$3.000
			Total :	**$444.800**

Tabla 4. Costo de componentes electrónicos.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Costo de materiales.

Ítem	Materiales	Cantidad	Costo unidad	Costo total ítem
1	Acrílico transparente 5mm x 200mm x 300mm	4 un	$12.000	$48.000
2	Acrílico azul transparente 5mm x 200mm x 300mm	4 un	$ 12.000	$48.000
3	Acrílico azul transparente 3mm x 200mm x 300mm	4 un	$12.000	$48.000
4	Tornillos rosca fina	30 un	$150	$4.500
5	Tornillos rosca fina	30 un	$150	$4.500
6	Tuercas rosca fina	60 un	$150	$9.000
7	Tornillos rosca ordinaria	30 un	$150	$4.500
8	Pegante instantáneo	4 un	$5.000	$20.000
9	Lija 320	5 un	$1.500	$7.500
10	Silicona	2 un	$1.000	$2.000
11	Cloruro ferrico	2 un	$1.500	$3.000
12	Thinner	1 lt	$10.000	$10.000
13	Pintura dieléctrica verde	1 lt	$4.000	$8.000
14	Esponjilla de brillo	4 un	$1.000	$4.000
15	Ruedas de neopreno	2 un	$15.000	$30.000
16	Caster ball	2 un	$10.000	$20.000
			Total :	**$271.000**

Tabla 5. Costo de materiales.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

Teniendo el costo de cada componente y el material necesario para construir el prototipo robot, solo hace falta añadir el costo del trabajo realizado, como lo muestra la **tabla 6**.

Costos del proyecto		Valor
Costo de papelería y consultas en internet.		$180.000
Costo de componentes electrónicos.		$444.800
Costo de materiales.		$271.000
Costo de la construcción del prototipo.		$500.000
	Total :	**$1.395.800**

Tabla 6. Costos totales del proyecto.

Fuente: Goyeneche, C. (2016) Datos de estudio. Sogamoso. UNAD

6. CONCLUSIONES

La construcción de un proyecto trae consigo nuevas dificultades técnicas, estas mismas fueron dando forma a los conceptos consignados en este documento. Todo el trabajo fue realizado con herramientas que cualquier amante de la electrónica tendría en su casa. La construcción del prototipo demandó desarrollo de habilidad manual y paciencia. Gracias a esto pude afianzar conocimientos de elaboración y manejo de simuladores para construcción de circuitos electrónicos, habilidad para la construcción de tarjetas impresas o PCBs, manejo de herramientas manuales, deducción y desarrollo de la imaginación. Además de los conceptos teóricos de electrónica análoga y digital adquiridos durante la carrera de ingeniería electrónica.

En cada una de las áreas involucradas para el desarrollo del proyecto, se presentaron una serie de situaciones que nos permiten destacar lo siguiente:

En la realización de la tarjeta PCB, se usó el programa de simulación Proteus ISIS que evidencio la ventaja de permitir modificar durante el proceso de creación de los esquemas el circuito, además de permitir simular el comportamiento del circuito al cargar el programa .hex , este mismo programa sirve para la realización de las tarjetas PCB, este simulador permite tener una vista en 3D para mejorar la estética del montaje. Luego de obtener la tarjeta esta se lleva a la construcción en físico, donde se usó la técnica de transferencia de calor y degradación por químico.

En la construcción de la plataforma se tienen en cuenta las dimensiones de los componentes y el comportamiento de los sensores y los motores con reducción, ajustándolos a la cinemática diferencial. Se usó como material de estructura el acrílico traslucido, que permite marcar y trazar sin dificultad, además el acrílico es fácil de cortar y pulir, se puede unir con pegante instantáneo y tornillos.

En la programación del microcontrolador PIC16F877A cumplió con las expectativas que se plantearon al inicio del proyecto, demostró ser un componente electrónico de amplia aplicabilidad, de grandes características y excelente desempeño.

El uso de la pantalla LCD y el teclado, posibilitaron interactuar con el prototipo robot y verificar el poder del microcontrolador. El control sobre los motores, de acuerdo a las señales recibidas de los sensores. El detectar obstáculos y generar respuesta a este acontecimiento, llevando al prototipo a ser seguro.

El control remoto le da versatilidad y le permite ser operado a distancia, comportándose el prototipo como un manipulador. Hace falta mejorar la potencia de emisión evitar oscilaciones de desplazamiento.

El robot no tiene una plataforma muy elaborada en detalles estéticos, debido a que se trata de un prototipo de bajo costo; se puede maquinar las piezas en CNC o cortar con láser para que las piezas sean exactas, se pueden usar componentes de mejor calidad en próximos prototipos.

El prototipo robot cumplió con el objetivo de transportar los tornillos. Dentro de lo supuesto el almacenista puede atender sus actividades en bodega y el proceso de producción es más óptimo, gracias a la introducción del prototipo robot.

El prototipo robot mostró un comportamiento esperado, realizando recorridos de seguimiento de línea y ubicación espacial dentro de un entorno definido, advierte obstáculos en el recorrido y permite ser operado de forma remota.

BIBLIOGRAFÍA

- Téllez Barrera (2007) Todo sobre mini robótica. Club Saber Electrónica Internacional.
- Solaque Guzmán, Molina Villa, Rodríguez Vásquez (2014) Seguimiento de trayectorias con un robot móvil de configuración diferencial. Recuperado de http://web.usbmed.edu.co/usbmed/fing/v5n1/v5n1a3
- Enrique Palacios, Fernando Remiro, Lucas López (2008) Microcontrolador PIC16F84. Tercera edición, Rama.
- Eduardo García Breijo, (2008) Compilador C CCS y simulador Proteus para Microcontroladores PIC. Primera edicion, Marcombo.
- Erasmo Bustamante (1991) Curso de Electrónica General. Fascículo uno y cuatro. Primera edición. Electrónica Bushers.
- Rodríguez Pozueta (2015) Maquinas de corriente continua. Recuperado de http://personales.unican.es/rodrigma/PDFs/Maquinas%20cc.pdf
- Pro-Lighting (2015). Diodo emisor de luz. Recuperado de http://www.pro-lighting.com/1/pdf/led.pdf
- Data Sheet (2001) PIC16F87X Data Sheet Microcontrollers. Microchip Technology Inc.
- Robert Boylestad, Louis Nashelsky, (2003) Electrónica Teoría de circuitos y dispositivos electrónicos. Octava edición, Pearson.
- Universidad de Guadalajara, (2012) Robótica, http://proton.ucting.udg.mx/materias/robotica/
- Instalador de PIC C. http://www.filefactory.com/file/ah05849/n/CCS_PCWH_v4_032_Reg_Files_rar
- Siemens Solid Edge ST7 https://www.plm.automation.siemens.com/es_sa/academic/resources/solid-edge/student-download.cfm
- Seguimiento de trayectorias con un robot móvil de configuración diferencial http://web.usbmed.edu.co/usbmed/fing/v5n1/v5n1a3

ANEXOS

Video Conceptos básicos 1. https://www.youtube.com/watch?v=wXHkUGHVasI

Video Conceptos básicos 2. https://www.youtube.com/watch?v=UDnKbwTcRho

Video Prueba 0. https://www.youtube.com/watch?v=NV_bDajoNaY

Video Prueba 1. https://www.youtube.com/watch?v=Zxil3CRVx0Y

¿Qué es robótica?
http://carlosgoyeneche.blogspot.com.co/2010/10/que-es-robotica.html

Robótica Beam
http://carlosgoyeneche.blogspot.com.co/2010/12/robotica-beam.html

Construcción de mini robot
http://carlosgoyeneche.blogspot.com.co/2010/07/construccion-de-robot.html

Sigue líneas simplificado
http://carlosgoyeneche.blogspot.com.co/2010/07/sigue-lineas-simplificado.html

Seguidor de línea Zamir 1
http://carlosgoyeneche.blogspot.com.co/2010/09/seguidor-de-linea-zamir-1.html

Z2C2
http://carlosgoyeneche.blogspot.com.co/2011/06/z2c2.html

Robot Driver
http://carlosgoyeneche.blogspot.com.co/2012/06/robot-driver.html

Soldadura con estaño
http://carlosgoyeneche.blogspot.com.co/2011/03/soldadura-con-estano.html

Canal en youtube
https://www.youtube.com/user/xotaruxesky